Order No.: 490

Annotated Help 3 Programs

vol. I

main help	15	pages
init help	8	–
P1	8	–
algol	7	–
binin	3	–
binout, outparam	6	–
check, compress, setsum, list	15	–
clear, res, set	8	–

December 1967

; slip<
; STOP, CLEAR

[8.8.67 (2) contents of main help]

[Page: track: Track numbers in bracket refer to version with buffermedia

2-3	0-1	All external entries use track 0 and 1 which dumps registers and core if necessary. The actual entry is determined before main help is read from drum.
4	2	Standard content of core 0-9, part of the code for exit and an address print routine.
5-6	3(3-4)	Init medium which transforms an area word to a more convenient discription of an input medium in cell -6 to -2. It also used for initialisation of program call and then the description is placed in cell c-3 to 1c. If the area word describes a point on magnetic tape then the tape is positioned.
7	about 4(5)	Entry from track 1 and special treatment of all entries to help.
8	about 5(6)	The help input program which reads help information and stores it in the parameter buffer.
9	about 6(7)	The parameter interpreter which scans the parameter list and accordingly selects media and starts program call by means of init medium.
10	about 7(8)	Tables and character look-up for help input.
11	about 10(9)	Contains a few constants, a routine for reading next character from internal media and a routine for adjusting booked or work after certain aux. programs.
12	about 8(10)	Program call. Reads the program to core and checks the sum if wanted. core 0-9 is always initialised and core inhibition is removed if wanted.
13	9(11)	Search catalog. Contains several small routines for selecting and reading tracks. The main purpose is to search for a text in the help catalog.
14	10(12)	Contains textprint and several constants. If buffer media are treated then a routine is included for getting next word from buffer medium. Otherwise is the read internal routine loaded to this track.

]

```
b i=0,d55; outermost block
d55=2    ; version number
 d1=100  ; first track loaded
 d3=0    ; free kind
 d4=0    ; discblocks mod 1024
 d5=39   ; first free block
 d9=0    ; first track of final help
d11=38   ; parameter track
d16=294  ; image track 0
d17=3    ; reader
d18=512  ; by inhibit
d19=960  ; image group
d21=34   ; first catalog track
d22=1    ; no of 320 track drums
d23=4    ; no of catalog tracks
d32=1    ; work in free
d33=0    ; first work track if not in free
d34=0    ; no. of work tracks if not in free
d35=0    ; running kind of aux programs (may be changed later)
d36=0    ; aux programs in free         (may be changed later)
d41=1    ; buffermedia treated
d43=17   ; help alarm unit, includes typewr input
d44=17   ; help standard output unit, used by date print and most
         ; aux. programs (except binout and other with perforator output)
d46=0    ; first free block : 1024
d50=960  ; image group during loading
d52=0    ; discblocks : 1024
d53=400  ; block length, disc
d54=1    ; lib station
 d=0     ; debug:=false
         ; STOP, SUMa
[d32, d36, d41 and d are booleans with true = 1, false = 0]
i redefine
s        ; allows redefinition of preceding names
         ; STOP, CLEAR
< d      ; if debug then place help relative to d1 and image on 268.
d5=40d1, d9=d1, d11=38d1, d16=268, d21=36d1 >

 d8=d1   ; define intermediate help base
d13=80   ; core base of parameter track
d14=40   ; core base of text buffer
d42=k    ; define image during loading.

d37=10, <d41, d37=12>    ; last rel help track:= if buffermedia then
d12=d37+d37+d37+d37+d37  ; 12 else 10
d12=d12+d12+d12+d12      ; core base of help:=
d12=-d12-d12+34          ; - (last rel help track - 1) × 40 - 6;

d10=d16-294, d20=1       ; SLIP names: FBO, first slip in core.

         ; Prepare loading of aux. programs only. Changed by INIT HELP.
 d2=10   ; coreaddress for loading descriptions of aux programs
d39=1    ; aux mode:= aux only;

d45=4, <d32, d45=0>      ; work as output:= if work in free then
                         ; free area else max work;
d51=6, <d32, d51=5>      ; date word relative:= entry after work
```

```
b k=2d8, i=d12, c85, e20   ; begin help
b k=d8, i=0, a20, b10      ; begin dump program
                             [stored registers]     [actions]
i=1 [qq       , hh c25]    ; 0 exit address       ; jump to primitive input
    it  960   , pt   1     ; group, entry kind    ; ENTRY MODE 5:
< d22-3                    ;                      ; entrykind:= right or left;
    gg  1     , gk   2     ; 2 by , tk
×   qq        , gk   2     ; 2                    ; store group, by, tk
>   vy d18+d43, hv   a     ; 3 indicator          ; by inhibition;

b:  qq 64.9+29.15+60.21+18.27+20.33+36.39 ; tSUM ; inhibit pattern
    qq 64.9+57.15+36.21+49.27+55.33+53.39 ; t image
                                                  ; PROGRAMMED ENTRIES:
a:  gi  3             IPC ; 6 M, marks           ; store indicator
    gm  6             MPC ; 7 S                  ; store M
    gs  7             IOB ; 8 R, overflow        ; store s
    gr  8             MOB ; 9 Radr               ; store R
    ga  9     , tl  -39   ; 10 s during prim inp ; store Radr for overflow
    pm  s             IRB ; 11 R sign            ; RB:= f-marked call;
    gr  11    , hr  r1    ; 12 p                 ; store sign; store p;
    gp  12    , arn b     ; [work for lyn]       ; simulate return
    sr  -1    , tk  21    ; if core [-1] ≠ inhibition then goto dump core;
    hv  a1            NZ  ; if core [-1] ≠ full inhibition then
    hv  a2            NA  ; goto get help;
a3: arn 1b    , pm  -1    ; 17 write: save core [-1];
a5: ga  a4    , tk  10    ;   writetext(textword);
a4: sy  -1    , ck  -4    ;
    nc  0     , hv  a5    ;
    lyn -1    , gm  -1    ;   typechar; restore core[-1];
b1: bs  1     , hv  a6    ;   if SUM then begin store return s;
c25h:gs 10    , vyn 16d18 ; PRIMITIVE INPUT: select reader;
    tl  -6    , ca  0     ;   read and pack until 64 hole is read;
    ly  r4    , hs  r-1   ;   store instruction;
    gm  s3    t-1     M   ;   see binout for further description;
                          ;   end;
a6: ca  0     , hv  a2    ;   if char = 0 then goto get help;
    nc  17    , hv  a3    ;   if char ≠ < then goto write;
a1: vk  d19   , it  1     ; dump core:
    vk  d16   , it  40    ;   for i:= 40 step 40 until 960 do
    sk  0     , it  -1    ;   dump core[i] to core [i+39];
c54: ncn 24     , hh  a1  ;
    [core dumped]

a2: vkf 960   , vk  1d9   ; get help: [f to stop checksum]
    lk  40    , vkn 38    ;   read track 1 of help;
a8: ar  37    , pa  b1    ;   while -, b-marked do sum words;
    ar  2     D       LA  ;   SUM:= true;
[37] hv (a8)  Dt  1   NB  ;   if sum ≠ 0 then begin textword:= SUM;
    hv (a3)   Dt  -1  NZ  ;   goto write end;
    arn 1             IQC ;   exit:= left;

d=40, d:                   ; track 0
i=0                        ;
    qq        , hh  c25    ;   cell 0 on track, (address chain)
```

```
i=40
     lk  40d13 , ck  10    ;   read track 38; if entrykind = right hp then
     ca  0     , pi  12    ;   exit:= right;
     vk  d19   , nc  1     ;   if entrykind ≠ programmed call then
     ps (0)    , hvn a9    ;   begin addr:= r1; goto set exit end;

     arn 41d13 , ga  1     ;   stored group:= cell 1 on track 38;
     ck  10    , pm  42d13 ;   stored by-tk:= cell 2 on track 38;
     gm  2     , pm  a10   ;   if entrykind = hs 2 then
     ca  1     , cln 20    ;   begin M:= hs2 entries; goto hs entry end;
     ca  0     , hh  a11   ;   if entry kind = hs 1 then
     ca  -1    , pi  12    ;   begin M:= hs 1 entries; goto hs entry end;
     ps (9)    , arn a12   ;   if entry kind = overflow h then exit:= right;
a11h:hv  a9    , pp (7)    ;   addr:= stored Radr; goto set exit;
     bs  p512  Xt  551     ; hs entry: if stored s < 40 then
d7:  qq[balance], hv a13   ;   begin RB:= f - marked call;
     pm  p40d13        IRB ;         simulate return
     ps  p40d13, hr  r1    ;   end;
a13: gs  7     , ps  p1    ;   stored s:= return s; addr:= call s+1;
ck  -10               NRB ;   if f-marked call then shift entry word;

a9:  bs  s506  t   515     ; set exit: if addr < 6 ∨ addr > 9 then
     gs  49d13         MQC ;   image[9] := addr, exit;
a12:
c55:[gt  r     , ps  c42]  ; [loaded by main help]
i=i+1 [entry overflow]     ;   help entry:= Radr;
     ncn(c54)  , hv  a16   ;   if core dumped then
     vk  d16   , sk  40d13 ;   begin dump track 38;
     vk  25d16 , sk  1000  ;         dump core 1000-1023 and registers
a16: vk  960   , pm  b     ;   end;
     gm  -1            M   ;   inhibit core;
c53: vy  d43   , pp  1     ; READ AND CHECK HELP: select(alarm unit);
a14: pp  p1    , vk  pd9   ;   release by-inhibit;
     lk  d12-40t   40      ;   read help to core;
     bs  p-512d37 , hv a14 ;
a15: vkn d21  , ar pd12-d37;   check sum of help tracks;
     ar  1     D       LB  ;
     ar  2     D       LA  ;
     bsp  519d12-d37       ;
     pp  p1    , hh  a15   ;   if sum ≠ 0 then textword:= SUM;
[75] ck [balance], lk d13  ;
     hv (a3)   D t-1   NZ  ;     goto write end;
a10=i+1                    ;   read first catalog track;
c56:[vk  d11   , hv  sc41  ;   select param track; switch to help entry;
a10: hs 1/hsf1/hs2/hsf2    ; two words loaded by main help]
i=i+2
     qqf [checksum]        ;
d=80, d:                   ; track 1

e                          ; end dump program
```

```
b i=d12, a50, b10          ;   begin local help names. Load to track 2 of help
c=-86                      ;   predefine place of search track

[cell 0-9 in core after exit]
     it  -1    , it  -1    ; 0 OVERFLOW v: entry kind:= -2;
     it  0     , pt  1     ; 1 OVERFLOW h: entry kind:= -1;
< d22-3                    ;   HS 1: HSF 1: entry kind:= 0;
     gg  1     , gk  2     ; 2 HS 2: HSF 2: entry kind:= 1;
     vy d18+d43, vk  960   ; 3 save group, track and by;
×    pa  1     t   960     ; 2
     gk  2     , vy d18+d43; 3 select alarm unit and inhibit by;
>    vk  38    , sk  0     ; 4 save core 0-39 on track 38;
     vk  d9    , lk  0     ; 5 read track 0 of help; wait for track;
     vk [group], vk [track]; 6 EXIT TO CORE: select group and track;
     qq        N [hh-hv]   ; 7 [used by HP-patches]
     vy [core init] t -1d18; 8 release by inhibit;
     qq  10    , hv (r)    ; 9 final exit

     [Part of code for exit]
c59: pi  2     t  -3   NO  ; set O: NRA:= R overflow;
     ac  512   DV  512 LRC ;   if -, stored overflow = R overflow then
     ac  512   Dt  512 NRC ;   O:= -, O;
     pi (3)    , lk  0     ;   restore indicator;
     vk  d19   , it (86d13);   read exit to core 0;
     pa  6     , hh  0     ;   prepare restore group; goto read image;

b a8 [address print, PM]
d40: pa  ra1   t  511      ; clear count: count:= 0;
d24: tk  20                ; spacing and print 30-39:
d25: tk  10                ; spacing and print 10-19:
d26: bt (ra1)  t   1       ; spacing and print  0- 9:
     sy        , hv  r-1   ;
d27: pa  ra1   t  506      ; COUNT 5: count:= 5;
d28: hs  ra1               ; SIMPLE PRINT:
     srn(ra4)  Dt  -16     ;
     hv  ra                ;
a1:                        ;
d29: qq[anslag], ck        ; print with zero as empty:
d30: bs 570 [case], it 510 ;
     sy (r-1)  , tl  -30   ;
     pa  ra7               ;
     dk  ra2   XV          ;
a6:  sy        , it  16    ;
a3:  pa  ra4   , it  -128  ;
a7:                        ;
d31: bt 1 [slip boolean], hrn s1    ;
     mln ra5   , tk  30    ;
a4:  ar  0     D       LZ  ;
a:   hh  ra3           LZ  ;
     qq (rd29) t   1       ;
     ga  ra6   , hv  ra6   ;
b:a5:m 10                  ; [base of constants]
a2:  9999                  ;
e

d=40d12, d:                ; track, adress print
```

```
b a10, b21
c28: c63=k-d8+d9          ;   define init medium track.
b20: vk  960   , vk  1c63 ; INIT MEDIUM: get second track of init medium;
     [blocks in area]
     lk (s-1)  , vk  960  ; ENTRY NO GET TRACK: R contains an area word;
[2b20]gp ra    , pp  c-1   ;   save p;
[3b20]bs 1     , pp  -4    ;   p:= if medium then -4 else search work;
     [medium]
     gr  rb3   , tl  -7    ;   area word:= R;
     hv  ra1          NT   ;   if kind ≥ 0 then goto internal;
     nc  -1    , pp  c22   ;   if kind ≠ ly then p:= selected output - 2;
     tk  17    , ga  rb10  ;   work:= core[p+2]:=
     gt  rb10  , pm  rb10  ;   vy <bits 10-19> t <bits 20-29>;
b10: vy  -1    t   -1      ; work for ly and sy media, and for label check
     gm  p2           M   ;   select medium;
a:   pp  -1    , hrn s1    ;   restore p; return normal;

a1:  ga  rb1   , tl  -25   ; INTERNAL: store kind;
     tln 16    , gr  rb20  ;   blocks:= bits 8-23;
     arn rb4   , gr  p1    ;   char count:= 1;
b1:   [kind]
<d41, ncn -1   , hv  ra2   ;   if kind ≠ drum ∧ buffer media then goto buffer;
×     ncn -1   , hv  c57   ;   if kind ≠ drum then goto kind alarm;
>    tl  -23   , ud  ra8   ;   medium word:= bits 24 - 39,;
     pm  d14-1 D           ;
a8:  gm  p5    t   -3  MA  ;   word address:= qq <text base - 1>,;
b4:  qq  1     , hv  ra    ;   restore p; return normal;

[buffer medium state]
c82:
b3:  qq                   ; area word. END OF INIT WITHOUT BUFFER.
<d41,qq     1.21          ; block increment
[2b3]qq d53.9   +7.39     ; current block
[3b3]qq        d53.39     ; block length
[4b3]qq  16    , il  -1   ; check block, read block
[5b3]qq          1.39     ; area word increment
b2:  qq     1.19+7.39     ; transfer upper
c29:
b21: qq [b3.9]6.19+1.39   ; put state
b13: qq       1.19        ; get label
b19: qq         1543.39   ; base for program buffer
c80:
b15: tcattap;             ; tape label
     qq 6.19+1544.39      ; get progr state

a2:  tln 4     , ck  -10  ; BUFFER:
     ga  rb5   , ac  r4b3 ;   unit:= bits 24-27; check block:= 16 + unit;
     arn c64   D          ;   save s;
     gs  ra3   , ck  10   ;   medium word:= get word track b-marks;
a4:  gr  p5    t   -3  MB ; [s-2]  read block:= unit;
b5:  it [unit] , pt  r4b3 ;
b6:  hs (s-1)  , udn ra4  ;   call second track; word address:= 0b;
a3:  ps  -1    , hv  ra   ;   restore s and p; return normal;

d=40b20, d:               ;   init medium 1
```

```
b1=b1-b6, b2=b2-b6, b3=b3-b6, b5=b5-b6, b10=b10-b6 ; define s-rel
b13=b13-b6, b15=b15-b6, b19=b19-b6, b21=b21-b6, b20=b20-b6;

[rel 0] gp ra9 , pp (sb1)  ; SECOND TRACK: save address of medium table;
     can p-2   , hh  ra5   ;   p:= kind; if kind = carrusel then goto carr;
     ca  p-3   , hv  ra6   ;   if kind = tape then goto tape;

     tln 12    , tk  18    ; DISC: current block := current bloack + first block.21;
a5h: hv  ra7   , tln 2     ;   goto finish;

     gr  s5b3  , tk  9     ; CARR: area word increment:= grouped;
     gr  s3b3  , tk  11    ;   blocklength:= 512 × grouped;
     gt  s2b3  , tk  10    ;   current block 10-19 := grouped;
     gr  s1b3  , tl  40    ;   block increment:= grouped.9;
     ga  s2b3  , hvn ra7   ;   current block 0-9:= reel and block; goto finish;

b7:  qq  400.19+7.39       ;
a6:  arn rb7   , gr  s2b3  ; TAPE:
     tk  -20   , gr  s3b3  ;   current block:= qq 400.19+7.39; block length:= 400;
     qq (s4b3) t   144     ;   check block:= unit + 160;
     pp (sb5)  , usn p64   ;   p:= unit; rewind(unit);
     arn sb13  , il  p     ;   read block (unit) length: (1) to: (0);
     ar  sb10  D           ;   read eof (unit);
     il  p64   , il  0     ;   get label to work;
     arn sb15  , sr  sb10  ;   if label ≠ ↯<cattap|then
c81: hh (s1)   t   1   NZ  ;   return error;
     tln 5     , ck  -10   ;   files:= bits 28-32;
     ga  rb17  , tln 37    ;   blocks:= bits 33-39;
     ga  rb18  , grn s1b3  ;   block increment:= 0;
b17: bt  -1    t   -1      ;   for i:= 1 step 1 until files do
     il  p64   , hv  r-1   ;   read eof (unit);
b18: bt  -1    t   -1      ;   for i:= 1 step 1 until blocks do
     iln p     , hv  r-1   ;   read block (unit) length: (0) to: (0);

a7:                        ; FINISH:
     sr  s1b3  , ac  s2b3  ;   current block := current block - block increment;
     arn s5b3  , sc  sb3   ;   area word := area word - area word increm;
     can(s3b20), arn sb19  ;   if -, medium then begin
     ac  s2b3  , ac  sb21  ;     current block:= current block + program buffer;
a9:  pp  -1    , gp  sb2   ;     put state:= put state + program buffer;
     ar  sb2   , ar  s3b3  ;     transfer upper:= transfer upper + program buffer
     gr  p-2   , it  sb3   ;   end; p:= address of medium table;
     pa  sb21  , arn sb21  ;   core[p-2]:= p.9 + transfer upper + block length;
     us  0     , hh  s     ;   put state to buffer; return to first track;

× c29=i, c80=i, c81=i>     ;   define c29, c80, c81 in no buffer case
e
```

```
c51: lk  d13   , pp  d13   ; RETURN FROM AUX PROGRAM:
     vk  d11   , arn d13   ;   read parameter track;
     ca  0     , hv  c46   ;   if -, hs 1 then goto select medium;
     ca  1     , pp  e12   ;   if hs 1 ∧ no interprete then param:= exit;
a41: vy  d44   , hv  c47   ; prep interprete: select (std output); goto interprete;

c49: pp  e11   , hv  a41   ; ENTRY HSF 1: param:= hsf 1; goto prep interprete;

c50: pa  d13   t   1       ; ENTRY HS 1: hs 1:= true;
     arn 49d13 , ga  r1    ;
     pp  -1    , vk  d19   ;   p:= exit addr; select image group;
     ps  d16   , pp  p-40  ;
     bs  p552  t   551     ;   p:= track rel (exit addr);
     ps  s1    , hh  r-2   ;   s:= track no(exit addr);
     pp  p80d13, vk  s     ;   param:= address(first param) - 1;
     lk  40d13 , ncns-25d16;   read track(s); if s ≠ image track 25
     vk  s1    , lk  80d13 ;   then read track (s + 1);
     vk  960   , hv  a41   ;   goto prep interprete;

c40: it  39[p] , pa  b3    ; ENTRY HSF 2:
c41: sy  64    , sy  29    ; ENTRY HP-BUTTON: writecr; writered;
     sy  58    , arn d13+1d51; count 5 and print (run);
b3:  sy  0     , hs  d27     ;   writechar (if hsf 2 then p else space);
     vy  d44   , arn d13+d51 ;   select (std output);
     ck  10    , hs  d26   ;   spacing and print (day);
     arn d13+d51, ck 20    ;   count 5 and print (month);
     sy  59    , hs  d27   ;
     arn d13+d51, ck -10   ;   simple print(year);
     sy  59    , hs  d28   ;
     arn 1     D           ;   run:= run + 1;
     ac d13+1d51, sc 39d13 ;   compensate catalog check sum;
     vk  d21   , sk  d13   ;   write catalog track;
     vk  d21   V           ;   skip line
c42: qq  e1    , hs  c23   ; ENTRY OVERFLOW: writetext (‡<overflow›);
     sy  0     , sy  0     ;
     can(c54)  , sy  53    ;   if core dumped then writechar(e);
     sy  56              LQB ;   if exit = right then writechar(h);
     arn 49d13 , hs  d28   ;   simple print(exit addr);
c62=i-1                    ;
c45: pm  e10   , vy  d43   ; SET TYPEWR: select (alarm unit);
     gm  -2              M ;   current medium:= vy 1 t 1016;
                           ; SELECT MEDIUM: ENTRY HS 2:
c46: grn d13   , arn -2    ;   hs 1:= false ; write param track;
     nc  1     , hh  a23   ;   if current medium = type wr then
     sy  62    , sy  58    ;   begin write black; writelc; writecr end;
a23h:sy  64    , vy  d44   ;   select(std output);
     hv  a40             NC ;   if current medium -< internal then
     pmn -2    , hs  c3    ;   begin select medium track;
     lk  d14   , vk  960   ;     read medium track;
     pa  b1    Vt c44  M   ;     read and look-up:= internal
a40: ud  -2                ;   end else
     pa  b1    t  c43  NC  ;   begin select(current medium);
                           ;     read and look-up:= lyn D end;
```

```
b k=d8, i=0                ;   finish track 1 of help
c40=c40-c41, c42=c42-c41, c46=c46-c41 ; define relative
c49=c49-c41, c50=c50-c41, c51=c51-c41 ; entries
c45=c45-c41                           ;
i=c56                                 ;
     vk  d11   , hv  sc41             ; define aux return
     qq c50.9+c49.19+c46.29+c40.39    ; entry table
i=c55
     gt  r     , ps  c42              ; define overflow entry
c40=c40+c41, c42=c42+c41, c46=c46+c41 ;
c49=c49+c41, c50=c50+c41, c51=c51+c41 ; redefine
c45=c45+c41                           ;
e                                     ;

b6:                        ; HELP INPUT:
a13: pp  d13   , pi  512   ;   param:= begin of line; state:= start;
a15: bs  p-38d13, hv a16   ; CHECK: if param > 38 then goto full;
a:   hv  a10          LTA  ; NEXT: if end input then goto end input;
b1:  hs [c43 or c44]       ;   read and look-up;
[s+1]pmn -1    Xt   b4     ;
     tl  20           LOA  ;   Radr:= syntax[kind] shift state;
     tl  10           LOB  ;
     ga  r1    , mb  a11   ;   state:= bits 0-1; end input:= bits 2-3;
     pi  -1    , ga  r1    ;   action:= bits 4-9;
     hvn -1    t   a       ;   switch to action;

a1:  pp  p1    , grn 2c    ; BEGIN NUMBER: param:= param + 1;
     gr  p             MB  ;   work:= 0; list[param]:= 0, number;
a2:  arn c43   , pm  2c    ; NUMBER:
     ca  16    , hhn  r1   ;   Radr:= if char = 16 then 0 else char;
     tk  -30   , ml  b     ;   if number too big then goto full;
     hv  a16           NZ  ;   work:= work × 10 + Radr;
     gm  2c    , hv  a     ;   goto next;

a3:  arn p     , nc  0     ; SHIFT NUMBER: if part 1 [param] ≠ 0 then
c61:
a16: qq  e2    , hs  c24   ; FULL: alarmprint(‡<full‡);
     ar  2c    , ck  10    ;   list[param]:= (list[param] + work)
     gr  p     V           ;   shift 10; work:= 0; goto check;
a5:  arn 2c    , ac  p     ; END NUMBER:
     grn 2c    , hv  a15   ;   list[param]:= list[param] + work;

a4:  arn(c43)  D           ; SINGLE:
     gr  p1            MA  ;   list[param + 1]:= char; single;
     pp  p1    , hv  a15   ;   param:= param + 1; goto check;

a12: qq  e4    , hs  c26   ; ANNUL: writetext(‡<annul‡);
                           ;   goto set typewr;
a14: arn 15.3  D           ; filled: list [param]:= list[param] + end word;
a6:  ac  p     , pp  p1    ; BEGIN TEXT: param:= param + 1;
     grn p     V       M   ;   list [param]:= 0, text;
a7:  arn p     , ck  -6    ; PACK:
     ar (c43)  D           ;   if list [param] -< full then
     nc (c43)  , hv  a14   ;   goto filled;
a17: gr  p     , hv  a15   ;   list [param]:= (list[param] shift -6)
                           ;   + char; goto check;
a8:  arn p     , ck  -6    ; END TEXT;
     ar  10    D           ;   while list [param] -‡ full do
     ca  10    , hh  a8    ;   list [param] := (list[param] shift - 6)
     ck  6     , hv  a17   ;   + 10; goto check;
a9:  qq  e3    , hs  c24   ; SYNTAX ERROR: alarmprint(‡<syntax‡);
```

```
a10: grn p1           MC  ; END INPUT: list[param + 1]:= end list;
     hv  a18          NTB ;   if end input = 3 then
a11: qq  63    , vk  960   ;   begin
     [mask]               ;     read track 0;
     vk  d9    , lk  0     ;     goto primitive input;
     vk  960   , hh  c25   ;   end;

a18: pp  d13   , gp  a13   ;   param:= begin of line:= 0;
c47:                       ;
c47h:pp  p-1   , hs  c52   ; INTERPRET: get param; this will select group 0;
c58: qq  e5    , hs  c24   ;   if number ∨ single then alarm print (ǂ<param>);
     hv  c46   , ga  r1    ;   if end list then goto select medium;
     pi [bits] , vk  c63   ;   indicator:= area bits;
     lk  c28   , vk  960   ;   read init medium 1;
     pa  3c28          LPA ;   if program then medium:= false;
     tl  -7    , ca  -4    ;   if kind = constant ∨ kind = -3 then
c57: qq  e8    , hs  c24   ; KIND ALARM: alarm print(ǂ<kind>);
     ca  -3    , hv  c57   ;
     qq  40c28 [init 2],tl7;
     hs  2c28  , qq  d13   ;   call init medium; s:= parameter|base;
     hs  c24           NZ  ;   if init error then alarmprint(ǂ<label>);
     qq  e9    , pm1 c80   ;   if -, program then goto interpret;
     hh  c47           NPA ;   if buffer media then begin
<d41,pa  c66   t   c-2     ;   modify get word to take description
     pa  c67   t   c-3     ;   from c-3 to 1c.
     pa  c68   t   c-1     ;   get state:= get progr state;
     pa  c73   t   c-1     ;   get status to c-1.
a24h:gm  c65   , ps  s1    ;   end;
×
a24h:ps  s1                ;
>    pm  p2           IRC ;   move remaining parameters to
     gm  s            MRC ;   parametertrack;
     bs  s-39d13, hv a16   ;   if more than 39 params then
     vk  d11    V      LRC ;   alarmprint(ǂ<full>);
     pp  p1     , hh  a24  ;
     sk  d13    , pm  2c   ;   M:= programarea;
     arn 1c           IRC ;   RC:= marks of medium word;
     srn(c15)   , hv  c48  ;   R:= -spec word; goto program call;

c52: pp  p1     , arn p1   ; GET PARAM: param:= param + 1;
     hv  s2           LC  ;   switch to end list,
     hv  s1           LB  ;              number,
     hh  s1           LA  ;              single;
     pa  a19   t   e6      ;   if name then
     pmn c69   DX      IZA ;   begin addr free word:= addr area word;
     hs  c2                ;     search catalog (param);
c60: pa  a19   t   e7  NT  ; NOT FOUND:
     hs  c24           NZ  ;     if not found then alarm print(
a19: qq  -1    , arn p1    ;     if undef then ǂ<undef> else ǂ<catalog>);
     tl  -6    , ca  -1    ;     param:= end of current name;
     pp  p1    , hh  r-2   ;     R:= area word; switch to name;
     arn 2c    , hh  s2    ;   end;
```

```
c44: hs  c27              ; READ AND LOOK UP: if internal then read internal
c43: lyn       D          ;   else begin
     [char]               ;     char:= lyn; if tapefeed then
     ca  63    , hv  c43  ;     goto read and look up end;
     ga  a36   , mb  a37  ;
a37: sc  r3    , ga  r1   ;   shift:= char ∧ 7 m;
     [mask]               ;
     pm  -1    Xt   b5    ;   RM:= inputtable [char ∧ 103];
     pm (r-1)  t    4     ;
[r3]
a36: tl  -1    , cl  -6   ;   Radr:= RM shift shift;
     ca  1     , hv  a12  ;   if annul then goto annul;
     ca  10    , hv  a9   ;   if end medium then goto syntax;
     ca  4     , gp  b6   ;   if CR then begin of line:= param;
     ca  11    , pan r1   ;   if LC then skip:= false;
     bs [skip] , hv (s)   ;   if skip then repeat call;
     ca  12    , gpn r-1  ;   if UC then skip:= true;
     ca  0     , hv (s)   ;   if blind ∨ UC ∨ LC then repeat call;
     ga  s1    , hv  s1   ;   kind:= Radr; return;

b5:  [inputtable]
qq  2.3+3.7+3.11+0.15+3.19+6.23+6.27+0.31+6.35+6.39 ;
;   ,   4   8    x    0    u    y    x    -    m    ;
qq  3.3+3.7+3.11+1.15+9.19+6.23+6.27+0.31+6.35+6.39 ;
;   1   5   9    aa   <    ∨    <    x    j    n    ;
qq  3.3+3.7+10.11+8.15+6.19+6.23+0.27+0.31+6.35+6.39;
;   2   6   end       s    w    x    x    k    O    ;
qq  3.3+3.7+0.11+0.11+6.19+6.23+5.27+0.31+6.35+6.39 ;
;   3   7   x    x    t    x    ,    x    l    p    ;

qq  6.4+0.8+6.12+6.16+6.20+12.24+4.28               ;
;   q   x   æ    d    h    UC   CR                 ;
qq  6.4+0.8+6.12+6.16+6.20+0.24                     ;
;   r   x   a    e    i    x                      ;
qq  0.4+0.8+6.12+6.16+11.20+0.24                    ;
;   x   x   b    f    LC   x                      ;
qq  6.4+0.8+6.12+6.16+7.20+4.24                     ;
;   ø   x   c    g    .    63                     ;

b4=i-2, a1=a1-a, a2=a2-a, a3=a3-a, a4=a4-a, a5=a5-a ; define relative actions
        a6=a6-a, a7=a7-a, a8=a8-a, a9=a9-a          ; actions
[syntax table: each entry consists of state.1+end.3+relative action.9]

[state: text  number       start        underlined      input:]
qq 0.1+0.9    +1.11+0.19   +2.21+0.29   +2.31+a9.39  ;  2 space
qq 0.1+a7.9   +1.11+a2.19  +1.21+a1.29  +2.31+a4.39  ;  3 digits
qq 2.1+a8.9   +2.11+a5.19  +2.21+0.29   +2.31+a9.39  ;  4 CR
qq 2.1+a8.9   +2.11+a5.19  +2.21+0.29   +2.31+a4.39  ;  5 ,
qq 0.1+a7.9   +2.11+a9.19  +0.21+a6.29  +2.31+a4.39  ;  6 letters
qq 0.1+a7.9   +1.11+a3.19  +2.21+a9.29  +2.31+a4.39  ;  7 .
qq 2.1+a9.9   +2.11+a9.19  +3.21+0.29   +3.31+ 0.39  ;  8 _
qq 2.3+a8.9   +2.13+a5.19  +2.23+0.29   +3.33+ 0.39  ;  <

        a1=a1+a, a2=a2+a, a3=a3+a, a4=a4+a, a5=a5+a  ; redefine
        a6=a6+a, a7=a7+a, a8=a8+a, a9=a9+a           ;
```

```
<-d41+1, d=i, i=-26> ; if no buffermedia then load to last help track;

b a4                 ;

e:   m -1                    ;
a:   pa  -3    t   -4        ; next word: char count:= -4;
c75: nc  39d14 , hv  ra1     ;   if word address = end of drum buffer then
     srn re    , ac  -2      ;   begin medium word:= medium word + 1;
     hs  c16            M   ;     select next track; drum:= true;
     is (c75)  , lk  s-39     ;     read track to drum buffer;
     vk  960   , it  -39      ;     word address:= base drum buffer end;
a1:  pm (-5)   Vt   1  LA     ;   if drum then begin word address:= word address + 1
c76: hs  20d14               ;     M:= current word:= word in buffer end
     hh  ra2   , hv  c70     ;   else if -, get word then
                              ;   alarm print(¢<fault¢); goto unpack;

c27: bt (-3)   t   1         ; ENTRY READ INTERNAL: char count:= char count + 1;
     arn -5    , hv  ra      ;   if char count > 1 then goto next word;
a2h: pm  -4    , cln -6      ; unpack: current word:= current word shift -6;
     ck  -4    , gm  -4      ;   Radr:= core[s+1]:= next char;
     ga  s1    , hr  s3      ;   return;

<-d41+1, i=d>                ;   continue loading

e5:  tparam;                 ;   alarm texts
e6:  tundef;                 ;
e7:  tcatalog;               ;
e8:  tkind;                  ;
e9:  tlabel;                 ;
e2:  tfull;                  ;
c22=i-2                      ;
     qq [selected output]    ;

                             ; ADJUST SPECIAL: R = last block + 1, M = first block
                             ;   in = bits of input area. Catalog track is last search
c74: hv  -9             NTB  ;   if not special then goto initialise help;
     hh  a4             LQA  ;   if work ouptut then begin
     gm  3d14  , sr  3d14    ;     work:= first block
     pa  3d14  t 1.3+d32.5   ;     + bits and kind; R:= no of blocks;
     tk  16    , ac  3d14    ;     work:= work + no of blocks end
a4h: hv  a3    , sr  d14     ;   else begin
     tl  d3.7-16, arn 1d14   ;     M.16:= last block + 1 - first free block;
     ck  d3.7+8 , tl -d3.7+16;     ;     booked:= M.16;
     ck  16    , gr  1d14    ;   end;
a3:  pi  0     , hs  c8      ;   ZA:= false; sum track;
     sk  d14   , hv  -9      ;   write track to drum; goto initialise help;

e
```

```
[M:=area; R:=-spec word    ;   RC:= marks of medium word; init medium called]
c48:  qq       V       NT  ; PROGRAM CALL: if spec word ≤ 0
     hv  a25           NC  ;     ∨ -, marks 00 then
a26: cln -26 [neg]       ;     R:= -(if drum program then
     ga  e15           NRB ;     std entry + no of tracks.9 else
     cl  26    , srn e15   ;     std entry + 10.9);
a25: mt  a26   , ga  a30   ;   R:= -r; unpack no of tracks,
     gt  a35   , ck  20    ;   entry,
     ga  a31   , ck  10    ;   core start,
     ga  a29   , vk  960   ;   first relative track;
     vk  2d9   , lk  0     ;   initialise core 0-9;
d38: pmn 1c    , hs  c3    ;   select (first track); marks:= 00;
     [max core for program call]
a29: ncn -1    , hs  a32   ;   while first rel track > 0 do
     [first rel track]     ;   skip track (first rel track);
a30: ncn -1    , hs  a31   ;   while no of tracks > 0 do
     [no of tracks]        ;   read track(no of tracks);
     grn -1            NQA ;   if -, inhibit then release core inhibit;
     arn 3c            LQB ;   if sum ∧ program sum ≠ 0 then
c77: hs  c24           NZ  ;   alarm print(ǂ<sum>);
a35h:qq  e16   , hv [entry];   entry program;

a31: pp[core start]Vt 40   ; procedure read track(count);
a32: is (a31)  , pp  s40   ;   begin core start:= core start + 40;
<d41,hv  a33           NRB ;   entry skip track: if -, drum program then
     gp  a34   , pp  -40   ;     for i:= core start - 40 step 1 until core start-1
a27: hs  c71               ;     do begin core [i]:= get word;
c70: hs  c24           NZ  ;     if fault then
     qq  e13           IPC ;     alarm print(ǂ<fault|);
     gm (a34)          MPC ;
     can p1    , hv  a34   ;   end
     pp  p1    , hv  a27   ;
× c70: >                   ;   else
a33: hs  c16           LC  ;   begin if marks 11 then select next track;
     lk  p-40  , vk  960   ;     read track to (core [core start - 40]);
     gpn a34   , pp  -1    ;   end;
a34: ar  p0    , pp  p-1   ;
     ar  1     D       LB  ;   for i:= corestart - 1 step -1 until
     ar  2     D       LA  ;     core start - 40 do
     bs  p41   , hv  a34   ;     program sum:= program sum + sum(core[i]);
     ac  3c                ;   count:= count - 1; marks:= 11;
     hv (s)    Dt  -1  M   ;   return end read track;

c24: pa  -9    t c45-c41   ; ALARM PRINT: prepare help entry to set typewr;
[1c24]it -10               ;   prepare return to init help;
c26: pa  c72   t c62       ; ANNUL PRINT: prepare return to set typewr;
     vy  d43   , sy  64    ;   select(alarm unit); writecr;
     sy  29    , hh  c23   ;   writered; goto text print;

e16: tsum;                 ;   may be used late during program call.
e3:  tsyntax;              ;   the following 3 cells may be spoiled
e4:  tannul;               ;   during interprete of program.
e10: vyn 1     t   1016    ;   constant for set typewr.

<i+86, itotal length
>                          ;   check help length
```

```
i=-86                       ; load to help track d8+d37-1
     [c-3 to 1c are used during init medium in program case]
c:   qq  d18.9+d17.9+c53.19+c45.29-c41.29+d53.39; constants for
     qq  d37.9                                  ; system punch
d6:  qq [area word]         ; check sum, work for number read
     qq[area word 1] ;
                            ;   comment found = NZA;
c1:  pmn c18   DX      IZA ; search: Raddr:= addr free word - 1; found:= f;
c2:  pm  rc12  , ga  rc17  ; get free: M:= catalog start; iparam:= Raddr;
c3:  ga  rc6   , pa  rc8   ; select: mode:= Raddr; reltrack:= 0;
     dln rc11  , ar  rc11  ;   group:= M : 960 + 960;
     ck  -10   , ga  rc5   ;   track:= M mod 960;
     cln -10   , ga  rc4   ;
c4:  is [track], can  s-960; select track: if track = 960 then
     pa  rc4   , it  1     ;   begin track:= 0; group:= group + 1 end;
c5:  vk [group], vk (rc4)  ;   wait track: vk(track);
c6:  can[mode] , hr  s1    ;   if mode = 0 then return
c7=10c1[10 used by Algol]  ; read track and sum:
c7:  lk  d14   , it  1     ;   to core (place 0); reltr:= reltr + 1;
c8:  qq [reltr], arn rc7   ; sum track: isum:= iword:= place 0;
     ga  rc9   , ga  rc15  ;
     vk (rc5)  , vkn(rc4)  ;   group vk(group); vk(track); R:= 0;
c9:  ar [isum]         IPC ; sumit: R:= set PC (store[isum]);
     ar  2     D       LA  ;   if LA then R:= R + 2 pos 9;
c10: sr  -1 [see c19-1]D LB;   if LB then R:= R + 1 pos 9;
     ar (rc8)  DVX     LC  ;   if -, LC then
     hv (rc9)  Dt   1      ;     begin isum:= isum + 1; goto sumit end;
c11: qq XVDN [=960.39]     ;   R:= R + reltr pos 9; R00:= R0;
c12:qqd21[catalog start].39;   if R = 0 then
     hv (rc15) Dt  -1  LZ  ;     begin iword:= iword - 1; goto get word end;
     sc (rc9)  , pm  rc13  ;   store[isum]:= store[isum] - R; M:= 1pos 9 + noise;
c13: hr s1[see i-1] X  LPB ; test end: if LPB then swap; return end;
c14: gp  rc19  , pa  rc20  ; cont search: itext:= p; equal:= t;
c15: pmn[iword]Xt   1  IPC ; get word: iword:=iword+1; M:= 0; R:= setPC(store[iword]
c16: hv (rc4)  Dt   1  LC  ; next track: test for last word on track;
     hr  s1    X       NZA ;   if LC then begin track:= track + 1;
                            ;   goto select track end;
c17: gr r[iparam]Vt 1  LA  ;   if found then begin swap; return end;
c18=c-1c17, c69=2c18       ;   if LA then areaword: store[iparam:=iparam+1]:= R;
     pa  rc17  V 2c18  LT  ;   if LA ∨ NT then nottext:
     pm rc10   , hv  rc13  ;     begin M:= -1.0+noise; goto test end end;
c19: sr [iname]t   1       ;   iparam:= addr are - 1; iname:= iname + 1
     pa  rc20  t 512   NZ  ;   R:= R - store[iname]; if R ≠ 0 then equal:= f;
     hv  rc15          NPB ;   if NPB then not last text word: goto get word;
c20: pi [equal, f=512]t 511;   found:= equal; goto cont search;
c21: qq d3.2+1.5, hv rc14  ;
     [value for test cancel allowed, used by algol cancel]

e                           ; end names a50, b10
```

```
b i=-46, a10, b5          ; load to help track d8+d37

c64=d9+d37                ; define get word track
c23: gs  ra    , an  s    ; TEXTPRINT: if called from half instr then
     is  s1          NA   ;     s:= address(s) else s:= address(s+1);
a1:  ps (s)    , arn rc23 ; repeat:
     pm  s     , ps  s1   ;   Radr:= next char;
a3:  cl  -6    , ca  10.5 ;
c72:                      ;
a:   ps  -1    , hr  s1   ;   if Radr = 10 then return;
     ca  15.5  , hh  ra1  ;   if Radr = 15 then get next textword;
     tk  -4    , ga  ra2  ;
     ca  63    , it  1    ;   if Radr = 63 then outchar(64) else
a2:  sy  -1    , hvn ra3  ;   outchar(Radr); goto repeat;
d15=i-c23                 ;   [length of text print used in slip]

e13: tfault;              ; e1 to e15 is used by get word for:
e1: b: k 29, 63           ; b: area word [rel 11 on track]
     toverflow;           ; 1b: block increment
     .                    ;
e14: vy  1     t  -1d18   ; 2b: current block
e11=i-1, e12=i            ; 3b: block length
     thsf1;               ; 4b: check block, read block
     texit;               ;
e15: qqf 10.9+40d13.19+40d13.29,; 5b: area word increment
     [standard entry]
<d41                      ;   if buffermedia then load get word;
     qq          1.39     ; 6b: 1
c65: qq  0.9+6.19+1.39    ; 7b: get state
c73: qq -4.9+1.19         ; 8b: get status
c71=-26                   ; ENTRY GET WORD:
c71: arn r6b   , ar (rc66) ;   [rel 20 on track] if 1+remaining words gq 0 then
c66: gr  -5    V       LT ;   goto get block;
     pa  rb1   , hv  ra4  ;   remaining words:= remaining words + 1;
c67: ar  -6    , il  0    ; transfer: get buf word(remaining words+trans upper);
c68: pm  -4    , hrn s1   ;   M:= current word; return normal;

a4:  it  rb    , pa  r7b  ; get block: repeat:= 0;
     arn r7b   , il  0    ;   get state to are word ... area word increm;
     arn r5b   , ac  rb   ;   area word:= area word + area word increm;
     arn r1b   , ac  r2b  ;   current block:= current block + block increment;
a5:  arn r2b   , ud  r4b  ; rep: read block(current block);
     arn r7b   , us  0    ;   put state back to buffer;
     arn r8b   , il (r4b) ;   wait for block; read status to current word;
     il  0     , arn(rc68) ;  if no error then
b1:  bt  0     Vt -100 LT ;   begin remaining words:= -block length;
     srn r3b   , hv  rc66 ;     goto transfer end;
     hh  s1               ;   repeat:= repeat + 1;
     hv  ra5              ;   if repeat gr 3 then error return; goto rep;
× c65=i, c66=i, c67=i, c68=i, c71=i, c73=i>
<i+9, i get word length
>                         ;   check length of get word
i=-9                      ;
     ps c51-c41, vk 960   ; INITIALISE HELP: s:= relative entry;
     vk  1d9   , lk  40   ;   read track 1 of help;
     vk  960   , hv  c53  ;   goto read and check help;
e                         ;   end names a10, b5
a                    ; STOP, SUM, a
i main help
s
```

[STOP, CLEAR]

[This tape may be loaded after main help and before aux. programs. The loading will then create a new help system including the loaded aux programs. The final storing of the aux. programs is controlled by means of the names d35=aux kind and d36=aux reserved.

Main help and aux programs are loaded to track d1 and onwards. Init help, special initialisation code and a primitive catalog is loaded to the core image. When Init help is executed the following takes place:

1) Special init code is executed.
2) Main help is initialised and moved to group 960 track d9 and on.
3) The primitive catalog is scanned and the entries are treated thus:
   All items are checked for proper kind if the reserve bit is set.
   Program items are moved to final place, the are word and sum word (if present) are adjusted. Moving depends on kind.
   Programs in free are moved to first free block and on, the free area is adjusted.
   Other drum programs are moved to group 960 and displaced d9-d8 tracks.
   Other disc programs are moved to block 0 and on.
   Carroussel programs are moved to reel 0, block 0 and on. Blocks grouped 3 per transport. Only one reel can be loaded.
   Tape programs are moved to the magnetic tape on station d54. A ↑<cattap↑ label is put on the tape and the last program terminted with EOF.
4) Core 0-9 are initialised.
5) The catalog is finished and placed on drum.
6) hsf 2 is performed.

   If Init Help is omitted then only the aux programs will be loaded to d1 and on. Calls of set, move and setsum will be loaded to the core image.]

```
b k=d42, i=0, a30, b30, e20 ; load to image
i=10                        ;
     hvf a                  ; execute special init; security end of cat.
b1=i, d49=b1, i=80i         ; define base of track buffers

b:  m 1                          ; constants and work locations
[1b]  960                        ;
[2b]  40                         ;
[3b]  qq 15.5+15.11+15.17+15.23  ; EOF constant
[4b]  qq 3b.9+1.19+4095.39       ; EOF to buffer
[5b]  qq [buf rel]               ;
[6b]  qq 1.19+4095.39            ; EOF or label to tape
[7b]  qq b1+40.19+41.39          ; 40 words to buffer
[8b]  qq 13b.9+1.19+4095.39      ; label to buffer
[9b]  qq [work and blocks]       ; blocks in primitive catalog
[10b] qq 9b.9+1.19+0.39          ; get status
c78:                             ;
[11b] qq [sum]                   ; sum of outputted words
[12b] qq [first block]           ; first block in primitive catalog
[13b] tcattap;                   ; tape label
[14b] qq d9.39                   ; to drum base
[15b] qq d8.39                   ; from drum base
[16b] qq 2d38.39                 ; max core for program call

e5:  vy  16    , sy  64    ; ALARM: writecr; writered;
     sy  29    , pp (s)    ;
     hs  a14   , ps  b3    ;   writetext(param); prepare return to scan cat;
a12: arn(b3)   t   1        ;
     hv  a12           NT  ;   scan catalog for first following text;
     pp (b3)   V       NC  ;   writetext(catalog)
     hv  a12               ;
a14: pmn p     X           ; writetext:
a22h:cl  34    , ck  -4    ;   Radr:= next char;
     ga  a13   , ca  15    ;   if Radr = 15 then
     pp  p1    , hv  a14   ;   write next word;
     ca  10    , hh  s     ;   if Radr = 10 then return;
a13: sy  -1    , cl  -6    ;   write char;
     hh  a22               ;   get next char;

b17: tmove trouble   ;     ;
b18: tformat   ;           ;
b19: tprogram call   ;     ;
b21: tkind   ;             ;
b23: tcat length   ;       ;
```

```
b13:  qq d53.39           ; PARAMETERS TO FREE: block length
[1b13]qq d46.29+d5.39     ;   first free block
[2b13]qq 8.27             ;   disc unit = 8

b14:  qq d53.39           ; PARAMETERS TO DISC: block length
[1b14]qq 0.39             ;   first disc block
[2b14]qq 8.27             ;   disc unit = 8
[3b14]qq d53.9+41.39      ;   d53 words to disc
e3:
a4h: grn 5b    , arn p1   ; END DISC: buf rel:= 0;
     tk  18    , ar  3b14 ; rep: write to disc(d53) words to : (first block);
     us  24    , iln 24   ;
     arn 10b   , il  0    ;   read status to work;
     arn 9b               ;   if error then goto rep;
     hh  a4          LT   ;
     arn b     , ac  p1   ;   first block:= first block + 1;
     hr  s1               ;   return;

b15:  qq 512.39           ; PARAMETERS TO CARR: block length
[1b15]qq 0.39             ;   first carr block, reel 0, block 0
[2b15]qq 7.27+3.29        ;   carr unit = 7, grouped = 3
[3b15]qq 1.19+41.39       ;   3 blocks to carr
[4b15]qq 0.39             ;   base next carr program, set in carr area
[5b15]qq 512.39           ;   physical block length, used in carr area
[6b15]qq 1.39             ;   increase to first carr block

e10: arn 5b    , ar 7b    ; END CARR: output track to (buffer [41+blockrel]);
     us  0     , arn 5b   ;   bufrel:= bufrel - block length;
     sr  b15   , gr  5b   ;   the remaining track words will be outputted
a2:  arn p1    , tk  30   ;   upon return;
     nc  15    , ca  14   ; rep: if first block -< 14, 15 then
a21: qq  b17   , hs  e5   ;   alarm(¢<move trouble¢);
     ar  3b15  , us  7    ;   write to carr(3) blocks to: (first carr block);
     iln 23    , arn 10b  ;   read status to work;
     il  0     , arn 9b   ;
     hv  a2          LT   ;   if error then goto rep;
     arn 6b15  , ac  p1   ;   first block:= first block + 3;
     hr  s1               ;   return;

b16: qq 400.39            ; PARAMETERS TO TAPE: block length
[1b16]qq 0.39             ;   first tape block
[2b16]qq d54.27           ;   tape unit = d54
[3b16]qq 400.19+41.39     ;   400 words to tape
```

```
e11: arn p1                ; END TO TAPE:
     hv  a19          NZ   ;   if first tape block = 0 then
     arn 8b    , us  0     ;   begin buffer[4095]:= tprogr;
     usn 64d54 , arn 6b    ;     rewind tape;
     us  d54   , iln 160d54;     write to tape (buffer[4095]);
     arn 10b   , il  0     ;
     arn 9b    , vy  17    ;     if error then
     hv  a30          NT   ;     begin
     sy  64    , sy  54    ;       writetext(¢<fault¢);
     sy  49    , sy  20    ;
     sy  35    , sy  19    ;       typechar; goto end to tape
     ly  9b    , hv  e11   ;     end;
a30: arn 4b    , us  0     ;     buffer[4095]:= EOF;
     arn 6b    , us  144d54;     write EOF on tape;
a19: grn 5b    , arn 3b16  ;   end;
a20: us  d54   V           ;   buf rel:= 0; write to tape(buffer[41:440]);
     arn 3b16  , us  32d54 ; rep: if error then skip 5 inches and write block;
     iln 160d54, arn 10b   ;   read status to work; back up if error;
     il  0     , arn 9b    ;
     hv  a20          LT   ;   if error then goto rep;
     arn 6b    , us  144d54;   write EOF on tape;
     iln 32d54 , arn b     ;   back up;
     ac  p1    , hr  s1    ;   first block:= first block + 1; return;

e1:  hs  e4                ; TRACK TO FREE DRUM: sum track;
     pmn p1    , dl  1b    ;
     ar  1b    , ck  -10   ;   group:= first free : 960 + 960;
     ga  r1    , cl  -10   ;
     vk[group] , ga  r1    ;   track:= first free mod 960;
     vk[track] , sk  b1    ;   write track (track buf);
     arn b     , ac  p1    ;   first free:= first free + 1;
     vk  960   , hr  s1    ;   return;

e2:  hs  e4                ; TRACK TO BUF: sum track;
     arn p     , sr  5b    ;
     sr  2b                ;   if buf rel + 40 > block length then
     hs (b11)         LT   ;   call (end action); will change buf rel;
     arn 5b    , ar  7b    ;   output track to (buffer[41 + buf rel]);
     us  0     , arn 2b    ;   buf rel:= buf rel + 40;
     ac  5b    , hr  s1    ;   return;

e:   hs  e4                ; OUT DISPLACE: sum track;
     is (b2)   , vk  sd9-d8;   select(track read + displacement)
     sk  b1    , vk  960   ;   write(track buf)
     hr  s1                ;   return;

c79:                       ;
e4:  gp  a5    , ppn 40    ; SUM TRACK: save p;
a6:  pp  p-1   , ar  pb1   ;
     ar  2     D       LA  ;   for p:= 39 step -1 until 0 do
     ar  1     D       LB  ;   sum:= sum + track buf[p] + marks;
     bs  p     , hv  a6    ;
a5:  pp  0     , ac  11b   ;   restore p;
     hr  s1                ;   return;
```

[START ACTIONS]

```
e6:  arn 15b   , sr  14b   ; DISPLACE AREA: if to drum base > from drum base
     hv  a21          LT  ;   then alarm(¢<move trouble¢);
     sc  12b   , pm  12b   ;   first block in catalog:= first block + to drum base
     hv  s1                ;   - from drum base; return;

e7:  arn 12b   , sr  p1    ; AREA IN FREE DRUM: if first free block > first block
     hv  a21          LT  ;   in prim catalog then alarm (¢<move trouble¢);
     pm  p1    , hv  s1    ;   first block in catalog:= first free block; return;

e8:  pm  9b    , arn p     ; BUF AREA:
     sr  b     , ml  2b    ;   blocks:= (blocks im prim catalog × 40)
     dl  p     , gr  9b    ;   + block length - 1) : block length;
     arn p1    , ar  p2    ;   first block in catalog:= first block + unit;
     hr  s1    X           ;   return; may be called from carr area;

e12: pm  4b15  , gm  p1    ; CARR AREA: first carr block:= base next carr progr;
     hs  e8                ;   call(buf area);
     arn 9b    , ac  4b15  ;   base next carr progr:= base next carr progr
     hv  s1                ;   + blocks; return;

d47:                       ; ENTRY AFTER SPECIAL INIT:
     vk  d50   , vk  d8    ; INIT TRACK 0-1:
     lk  b1    , vk  1d8   ;   read track 0 and 1;
     lk  40b1  , vkn 960   ;
a7:  ar  37b1              ;   R:= sum of words 37 to 79 + sum of marks;
     ar  2     D       LA  ;
     hv (a7)   Dt  1   NB  ;   track 1 sum:= track 1 sum - R;
     sc  79b1  , ar  79b1  ;   R:= R + track 1 sum;
     arn 512   D       NZ  ;   if overflow then begin
     ac  d7+b1 , ac  75b1  ;   balance := balance + 512.9;
     vk  d9    , sk  b1    ;   track 1 sum := track 1 sum + 512.9 end;
     vk  1d9   , sk  40b1  ;   write track 0 and 1;
a8:  vk  d50   , it  1     ;
     vk (b2)   , lk  b1    ;   for track read:= track 2 of help step 1 until
     vk  960   , hs  e     ;   last main help track do
     bt  d37-2 t   -1      ;   begin read track (track read);
     hv  a8                ;   out displace end;
     vk  d9+d37-1 , lk b1  ;   read search track;
     vk (r-1)  , arn 11b   ;   help sum:= help sum - sum;
     sc  d6-c+b1, sk b1    ;   write search track;
```

```
b3:  arn b4    t   1       ; SCAN CAT: current:= current + 1;
     hv  a23           LC  ;   if cat[current] a-marked then goto area;
     hv  a9            LA  ;
     hv  a10           NC  ;   if - - - 0-marked then goto text;
     hv  a11           LZ  ;   if - - - =0b then goto end scan;
a23: qq  b18   , hs  e5    ;   alarm(¢<format¢);

a10: tl  -6    , nc  -6    ; text: if cat[current] -¢ end text then
     hv  b3                ;   goto scan cat;
     acn(b3)           MB  ;   b-mark (cat[current]);
     is (b3)   , arn s1    ;   if cat[current + 1] -¢ specification then
     qq        V       LT  ;   begin
     hv  a24   X       NC  ;     if spec required then
b20: bs  0     , hv  a25   ;     alarm(¢<program call¢);
     [spec required]       ;      goto scan cat;
     hv  b3                ;   end;
a24: xrn       , cl  10    ;
     ck  20    , tk  30    ;   if tracks read to core × 40
     ck  10    , ml  2b    ;   + core address for first track
     xr        , sr  16b   ;   < 2 + max core for call then
     hv  b3            LT  ;   goto scan cat;
     hv  b3            NPA ;   if -, program then goto scan cat;
a25: qq  b19   , hs  e5    ;   alarm(¢<program call¢);

a9:  ga  r1    , tl  -7    ; AREA: indicator:= area bits;
     pi  0     , nc  d3    ;
     hv  a23           LPB ;   if reserved ∧ kind ≠ free kind then goto alarm;
     ga  b26               ;   store kind
     ga  b5    , tl  -25   ;
     tln 16    , gr  9b    ;   work:= tracks := no of blocks;
     ck  -10   , ga  b6    ;
     tln 16    , gr  12b   ;   track read:= first track:= bits 23-39;
     ck  -10   , ga  b2    ;
     pa  b20               ;   spec required:= false;
     hv  a15           NPA ;   if -, program then goto end area;
     bs (b6)   t   19      ;   if tracks > 19 then
     pa  b20   t   1       ;   spec required:= true;
     ncn(b5)   , pa  b20   ;   if -, drum then spec required := false;
b5:  arn[kind] Vt  b7  NPB ;   R:= if -, reserved then action table [kind]
     arn b8                ;   else free actions;
     gt  b10   , gt  b24   ;
     ck  -10   , gt  b9    ;   unpack outtrack action, start action,
     ck  -10   , ga  b25   ;
     ga  b11   , gt  b22   ;   end action, parameter base.
b22h:grn 5b    , pp  0     ;   buf rel:= 0; p:= parameter base;
b9h: qq  b21   , hs  0     ;   call start action, prepare for alarm(¢<kind¢);
     arn(b3)   , cl  -16   ;
     pm  9b    , cl  -16   ;   cat[current]:= bits 0-7 + blocks.23
     ck  -8    , gr (b3)   ;   + first track.39;
     grn 11b   , vk  d50   ;   sum:= 0; select load image group;
b2:  vk  1d8   , lk  b1    ;   for i:= tracks step -1 until 1 do
     [track read]          ;   begin read track(track read);
b10h:vk  960   , hs  0     ;     switch to outtrack action;
     vk  d50               ;
b6:  ncn[tracks] t -1      ;     track read:= track read + 1;
     hv (b2)   Dt   1      ;   end;
     ps  40    , ps  s-1   ;
     grn sb1           M   ;   clear track buffer
     bs  s     , hh  r-2   ;
```

```
b26: can[kind] , hv  e9    ;   if kind ≠ drum then
b11: pa [end action] Dt b25;   begin end action:= end output
b24h:ps  r-1   , hv  0     ; rep: outtrack action; goto rep;
b25: hs [end action]       ;   call end action end;
e9:  srn 11b   VX          ;   M:= -sum; skip line;
a15: is (b3)   , pm  s1    ; end area: M:= cat[current + 1];
     is (b3)   , arn s1    ;   if cat[current + 1] a-marked then
     gm (b3)   Vt   1  LA  ;   begin current:= current + 1; cat[current]:= M
     hv  a23           NT  ;   end else if cat[current + 1] ≠ text then
     hv  b3                ;   alarm(↯<format↯); goto scan cat;

[Format of action table: Start action.9 adjust first block and blocks
which then are assembled in the central routine. Out track action.19 sums
and outputs one track, this may call the end action to output a block.
End action.29 is blind in case of drum program but outputs the last block
in other cases. Param base.39 points to 3 cells containing: block length,
first block, unit.]

[-4] qq e5                       ; constant: alarm
[-3] qq e5                       ; not used: alarm
[-2] qq e5                       ; sy-progr: alarm
[-1] qq e5                       ; ly-progr: alarm
b7:  qq e6 + e.19+ e9.29+  0.39  ; drum: displace area , out displace, blind, blind
[1]  qq e8 +e2.19+ e3.29+b14.39  ; disc: buf area      , track to buf, end disc, di
[2]  qq e12+e2.19+e10.29+b15.39  ; carr: carr area     , track to buf, end carr, ca
[3]  qq e8 +e2.19+e11.29+b16.39  ; tape: buf area      , track to buf, end tape, ta
b8: < d3                         ;
     qq e8 +e2.19+ e3.29+b13.39  ; free disc: buf area , track to buf, end disc, fr
×    qq e7 +e1.19+ e9.29+b13.39  ; free drum: area in free, track to free, blind, f
>                                ;

a11: arn 1b13  , ac  b12   ; END SCAN: free:= free + to free.39
     tk  16    , sc  b12   ;   - to free.23;
     vk  960   , vk  2d9   ;   initialise core 0-9
     lk  0     , vk  960   ;

a16h:pp  b12   , ps  0     ;   get:= address of free word;
     srn 3     Dt   1      ; fill cat track: track buf[39]:= rel track.9c;
     gr  39b1          MC  ;   for s:= 0 step 1 until 38 do
a17: arn p             IRC ;   begin track buf[s]:= cat[get];
     pin 1             LC  ;     if c-marked then
     gr  sb1           MRC ;     track buf[s]:= 0b;
     ar  1     D       LRB ;
     ar  2     D       LRA ;     track buf[39]:= track buf[39]
     sc  39b1              ;     - track buf[s] - marks;
     pp  p1    , ps  s1    ;     get:= get + 1;
     bs  s473  , hv  a17   ;   end;
a18: vk  d21-1 t   1       ;   cat track:= cat track + 1;
     sk  b1    , vk  960   ;   write(cat track);
     ps (b3)   , it  p512  ;   if get ≤ last scanned then
     bs  s513  , hh  a16   ;   goto fill cat track;
     pp  b23   , vy  16    ;   p:= alarm text;
     is (a18)  , bs s-d21-d23+1 ; if cat track - first cat track + 1
     ps  r1    , hv  a14        ; > no of cat tracks then alarm(↯<cat length↯)
     hsf 2                      ; call help
d2=i                            ;
a:   hv  d47                    ; prepare loading of special init
```

```
i=800                              ;
b12=i, b4=1i                       ; define place of primitive catalog
<d3                                ; free area:
qq d3.2+1.3+1.6+d52.13+d4.23+8.27,; special, inhibit, free blocks, first free
×                                  ; initially:
qq d3.2+1.3+1.6+d22.15+d22.17,     ; special, inhibit, free to, 0
>                                  ; finished in Init Help.
qq d23.7+d46.29+d5.39,             ; max free:
tfree;                             ; catalog tracks, booked, min first free

d19=d19-960                        ; work:
qq 0.2+1.3+d32.5,                  ; special, work in free, tracks, first track
<-d32+1                            ; if -, work in free then
qq d34.23+d19.29-d19.33+d33.39,    ; load max work: work tracks, first work
>                                  ;
twork;                             ;
qq -4.2+2.19+6.29+67.39,           ; date:
qq 0.9,                            ; constant, day, month, year
tdate;                             ; run number

qq 26.23+d19.29-d19.33+d16.39,     ; image:
timage;                            ;     26 tracks, first track of image
d19=d19+960                        ;

qq -1.2+d17.19+1016.29,            ; reader:
tr;                                ; ly medium, 0 or 3, bits 7-9

qq -1.2+1.19+1016.29,              ; typewriter:
tt;                                ; ly-medium, 1     , bits 7-9

qq -2.2+32.19+903.29,              ; perforator:
tp;                                ; sy-medium, 32    , bits 3-6

qq -2.2+8.19+903.29,               ; lineprinter:
tl;                                ; sy-medium, 8     , bits 3-6

qq -2.2+16.19+903.29,              ; writer:
tw;                                ; sy-medium, 16    , bits 3-6

qq -2.2+0.19+1023.29,              ; no-alarm unit:
tx;                                ; sy-medium, 0     , no bits active

d48=i, d39=0, d1=d1+1d37           ; prepare loading aux. programs
qqf                                ; aux only:= false;
e                                  ; end image block
[STOP, SUM]xi init help
s
```

```
;slip<

[7.8.1967                                                    P1         page 1]


[This program includes the Gier Algol 4 translator in the Help 3
 system, or punches a paper tape version of the translator. The
 translator should be placed on the drum before loading of this
 program. If wanted, the translator may be moved to another medium.
 The program enters a description in the catalog. The program
 requires the aux programs: res, set, move and setsum]


b a50, b30, c20, d50       ;
iP1, include algol:
d i=15, d37=0              ;

d24: qq [first track]      ;   first track of translator;
d i=i-1                    ;

itype: actual first track of translator
s                          ;

itype: d35 = kind of final translator medium
s                          ;

<-d35,d37=1>               ;   if kind less 0
<d35-6,d37=1>              ;    ∨ kind gr 6
<d35-3,<-d35+6             ;    ∨ kind gr 3 ∧ kind less 6
 d37=1                     ;   then begin message wrong kind; hsf 2 end;
>                          ;

<d37, iwrong kind
     hsf2                  ;
       ei-1                ;
>                          ;

c:   qq  1.39              ;
c5:  ttranslator medium;   ;
c9:  qq  40.39             ;
c10: qq  39.39             ;
c11: qq  400.39            ;
c12: qq  512.39            ;
```

```
a18: vy  17    , sy  29    ; alarm:   select(type writer);
     pt  a22   , hh  r2    ;   write red;   to help := true;  goto text;
a19: pt  a22   t   1       ; writetext:   to help := false;
a25: vy  17    , sy  64    ;   select(type writer);
     it (s)    , pa  r1    ; text:   writecr;
a20: pmn -1    X           ; next word:   M := 0;  R := next text word;
[1]  cl  34    , ck  -4    ;   RM := RM shift 34;
     ga  a21   , ca  15    ; next char:  R := R shift -4;  char := RADDR;
     hv (a20)  D   1       ;   if char=15 then goto next word;
     ca  10    , hh  a22   ;   if char=10 then goto end text;
     ca  63    , it  1     ;   if char=63 then char := char + 1;
a21: sy [char] , cln -6    ;   writechar(char);  R := 0;  RM := RM shift -6;
a22: hh  1a20  , bs[tohelp];   goto next char;
     vy  33    , hr  s1    ; end text:   select(type writer input and puncher);
     hsf 2                 ;   if to help then hsf 2 else return,

a12: hv (a14)  D   1       ; next track:    track := track + 1;  goto SELECT;
a13: dln a16   , ar  a16   ; to core:
     ck  -10   , ga  a26   ;   group := M:960 + 960;
     cl  -10   , ga  a14   ;   track := M mod 960;
a14: is [track], can s-960 ; SELECT:  if track = 960 then
     pa  a14   , it  1     ;    begin track := 0;  group := group + 1 end;
a26: vk [group], vk (a14)  ;   select(group);  select(track);
a15: lk  d2    , vk (a14)  ;   from drum;   wait drum;
     qq (a15)  t   40  LZA ;   if count base then increase track place;
     hr  s1                ;   return;
a16: qq  960.39            ;   960;

d20: gs  d21   , pmn d24   ; GET GP SEGMENT:
     hsn a13          IZA  ;   count base := true;
     arn d2    , ga  a1    ;   to core(first track translator);
a1:  pp [GPsize],vy  33    ;   p := size := part 1 of first word GP;
     pp  p-40  , ps  r-1   ;   for p := p-40 while p gr 0 do
     bs  p     , hv  a12   ;     next track;
     pa  a15   t   d2      ;   restore track buffer pointer;
     arn d9    , ga  r1    ;
     pp[reltab]t   d2      ;
<-d35+6                    ;
     gp  b1    , ck  21    ;
×    gp  a10   , ck  21    ;
>                          ;
×    gp  a10   , ck  21    ;   take abs address segm table;
>                          ;
d21: hv [old s]t   1   IOB ;   OB := GP name(e44);  return;


<-d35+6                    ;
b10: grn b2    , hs  d20   ; START INTERNAL:
[1]  pp  -1    , pp  p1    ;   blocks := 0;  GET GP SEGMENT;
b1:  pm p[base], ncn p-14  ;   for p := 0 step 1 until 16 do
     tl  9     , tln -9    ;     blocks := blocks + (if std proc code segm
     tln 19    , ac  b2    ;         then bits(0,19,segm table[p])
     bs  p496  , hh  1b10  ;         else bits(10,19,segm table[p]));
```

```
     arn d9    , gt  r1    ;
     arn 184   D   [e4]    ;   specword := (184+e4) pos 19
     ck  -10   , gt  b3    ;         + ((GP size+39):40) pos 9
     ck  -10   , ac  b3    ;         + (184+e4) pos 29;
     pmn(a1)   D X t -1    ;
     cl  10    , dln c9    ;
     ar  c     , ck  -10   ;
     ga  b3    , arn b2    ;   part 1 of specword := tracks in GP;
     sr  c     X           ;   move parameter 1 :=
     dln c9    , ar  c     ;     tracks in translator pos 23
     tk  16    , ar  d24   ;   + first track translator pos 39;
     gr  b9    , grn -1    ;   remove core inhibition;
     bs  511d35 V      LOB ;   if GP name(e44) = 1 ∧ d35 less 1
     bs  d35               ;      ∨GP name(e44) = 0 ∧ d36 gr 0
     qq  c5    , hs  a18   ;   then alarm(translator medium);
     arn b2    , sr  c     ;
     xr        , dln d13   ;
     ar  c     , gr  b2    ;   blocks := (blocks - 1):blocklength + 1;
     arn b2    , nc  39    ;
     acn(r-1)          MB  ;
     hv (r-2)  D t 1   LZ  ;

d d36=0                    ;
<-d35+2,iif reserved then type: d36=1
s>                         ;

<-d35+6                    ;
     hs  1                 ;
     hv  b5                ;
<d36,tres;                 ;
x<-d35+6                   ;
     tset;                 ;
     qqf d35.39            ;
>                          ;
<-d35+6                    ;
b2:  qqf [blocks]          ;   if reserved then res else (set,kind),
     qqf 3.39              ;
     qqf                   ;
     qqf                   ;   typein,

<-d35+1                    ; DRUM:
  d i=i-3, d13=c9          ;
<-d36+1,itype: final first track
  s                        ;
>                          ;

<d35,<-d35+2               ; DISC:
  d i=i-3                  ;
<-d36+1,itype: disc no, first block
  s                        ;
>                          ;

<d35-1,<-d35+3             ; CARROUSEL:
  d i=i-2, d13=c12         ;
  itype: reel, first block
  s                        ;
>                          ;

<d35-2,<-d35+4             ; TAPE:
  d i=i-3, d13=c11         ;
  itype: unit, file, block
  s                        ;
>                          ;
```

```
<-d35+6                    ;
     qq  39    ,           ;   p
     qq  57    ,           ;   i
     qq  52    ,           ;   d
     qqf 0                 ;   0

b4: itype: tname of translator;
    s                      ;

b3: qqf [specword]         ;   name,specword
    qqf 0,                 ;   <

b5:  arn b4-1  t   1   IRC ; move:
[1]  gr  b9    t   1   MRC ;
[2]  gr  b6    t   1   MRC ;   set name of translator as second parameter
     hv  b5            NRC ;
     grn (1b5)         MC  ;
     grn (2b5)         MC  ;

     hs  1                 ;   call move
     hv  b8                ;   and return to setsum;
     tmove;                ;
     qq  50    ,           ;   move, b
b9:  qqf [moveparameter 1] ;         actual place on drum,
     [name of translator]  ;         name of translator<

d i=50i                    ;

b8:  hs  1                 ;   call setsum
     hsf 2                 ;   and return to Help;
     tsetsum;              ;   setsum,name of translator<
d b6=i-1                   ;
     [name of translator]  ;

<d35,<-d35+2, itype: if CDC disc then 640 else 400
d13: qq [block length]     ;
     d i=i-1               ;
s                          ;
>                          ;

<-d35+6                    ;
d d2=50i, d9=1d2           ;
e b10                      ;
>                          ;
```

[The following code punches two paper tapes. The first tape consists of GP and pass1, the second of the other translator segments (passes). The sum of all segments is checked. The GP segment is modified before output: Track 5e14 (block for next segment 2) is replaced by the papertape version (see below d ff), and in tail of GP a code for writing GP on drum after input is inserted (see below d1 ff). The first tape may be read in by means of the auxiliary program algol. The code is only loaded if kind (d35) = 6]

```
b11: hs  d20               ;   GET GP SEGMENT;
     pp (a1)   V       NO  ;   if GP name(e44) ≠ 0 then
     qq  c5    , hs  a18   ;    then alarm(translator medium),
     hsn a23               ;   p := size;  R := 0;  sum;
     pmn(d2)  DXV t -1 LZ  ;   if R≠0 then alarm(pass sum);
     qq  c4    , hs  a18   ;
     cl  10    , dln c9    ;   GPsize := GPsize - 1;
     ck  -10   , ga  d23   ;   tracks := GPsize:40;
     pp (d2)   t   d4      ;   p := size := GPsize+init code length;
     arn  d9   , pa  d12   ;
     gt  d22   , ck  10    ;   part 2 of first word GP := code base
     ga  d1    , ga  r1    ;                               + old size;
     ps [e4]   , it  sd5   ;   set e13 and e4 in Get next segment 2;
d12: it  p0    , pt  d2    ;
     it  s21   , pa  d     ;
     it  s263  , pa  d13   ;   set e16-1 in Get next segment 2;
     it  pd6   , pa  a3    ;
     it  p     , pt (a10)  ;   part 2 of GP segm word := size;
     pp  d7    , vy  33    ;   select(typewriter and puncher);
     pm (d2)   D X         ;   words := size;
     ck  10    , gr  c7    ;
a2:  pm  pd           IRC ;   Move init code
a3:  gm  p0           MRC ;   and block for next segment 2
     pp  p-1   , can p-39  ;   to final places
     pa  a3    t   d8      ;
     bs  p     , hv  a2    ;
     pp (d2)               ;
     pp  p-1   , hsn a23   ;   p := size-1; R:=0; sum;
     pp (d2)   , gr  pd11  ;
     srn pd11  , gr  pd11  ;   last word GP := -R;
     pp  -1    , hs  a17   ;   p := -1; spaces;
     qq  c2    , hs  a19   ;   writetext(tear off. first tape...);
     it (a10)  , pt  a24   ;   base := base segm table;
```

```
a4:  lyn[char] D           ;   typechar;
a5:  hs  a17               ; PUNCH SEGMENT:
     sy  13    , grn c1    ;   spaces;  writetext(aa);  sum := 0;
a6:  pp  p1                ; NEXT WORD:   p := p+1;
     bs  p-39  V       NZA ;   if -,count base then begin
     hh  a7                ;   if p gr 39 then
     pp  0     , hs  a12   ;    begin p := 0; next track end;
     arn pd2               ;   M := R := core[track buffer base + p];
     ar  2     D       LA  ;   if mark a then R := R + 2 pos 9;
     ar  1     D       LB  ;   if mark b then R := R + 1 pos 9;
a7h: ac  c1    , pmn pd2   ;   sum := sum + R ;  R := 0; end;
     ar  2.2   D       LA  ;   if mark a then R := R + 2 pos 2;
     ar  1.2   D       LB  ;   if mark b then R := R + 1 pos 2 ;
     pa  a9    X   5       ;   swap;   RM := RM shift 32;
a8h: cl  32    , ga  r1    ;   for i := 1 step 1 until 6 do
     sy [char] , cl  -7    ;    begin writechar(bits(3,9,R));
a9:  bt  5     t   -1      ;    RM := RM shift -7
     hh  a8                ;    end;
     arn c7    , sr  c     ;   words := words - 1;
     arn c1    V       LZ  ;   if words ≠ 0 then goto NEXT WORD;
     gr  c7    , hv  a6    ;   if sum ≠ 0 then
     pin 0             LZA ;    alarm(pass sum);  count base := false;
a10: arn[segm] XV t1   LZ  ;   R := next segm word;
     qq  c4    , hs  a18   ;
a24: ns (a10)  , ps s[base];
     ncn s14   X           ;   if not std proc code segm word
     tk  10    , ck  -10   ;    then clear part 1 of R;
     tl  -20   , gr  c7    ;   words := bits(0,19,R);
     can s2    , hv  a11   ;   if pass2 segm word then goto NEXT TAPE;
     ncn s17   , hv  a5    ;   if more segments then goto PUNCH SEGMENT;
     pt  a22   , hs  a17   ;   spaces;
     qq  c6    , hs  a25   ;   writetext(tear off. end...);     hsf 2;
a11: hs  a17               ; NEXT TAPE:   spaces;
     qq  c3    , hs  a19   ;   writetext(tear off. next...);
     hv  a4                ;   goto PUNCH SEGMENT;

a23: pp  p-1   , ar  pd2   ; sum:  for p:=p, p-1 while p gr 0 do
     ar  2     D       LA  ;   begin R := R + word[track buffer base + p];
     ar  1     D       LB  ;    R := R + marks pos 9;
     bs  p     , hv  a23   ;   end;
     ck  0     , hr  s1    ;   clear R00;  return;

c1:  qq [sum]
c7:  qq [words]
c2:  ttear off. first tape is punched by typing SPACE;
c3:  ttear off. next tape is punched by typing SPACE;
c4:  tpass sum;
c6:  ttear off. end paper tapes;
```

```
d:   [Get next segment 2, paper tape version]

b a2, b10, c10, e28        ;
d e4=0,e13=21,e28=163e13   ;
d e16=80e28,e11=34e28      ;

c1:  gmn[e13]  V X     IZC ; Get segment:  e13 := R := M; goto fetch e4;
c2:  grn rb6               ; Get word:  segment transp := f; R := 0;
d22h:gs  rc8   , ps [e4]   ; fetch e4: save s := s;  s := e4;
     hh  rc5           LZ  ;   if R=0 then goto read start;
     gt  r     , pp [words];   sum ok := transport ok := t;
     ga  rb5   , tk  20    ;   p := bits(10,19,R); first core := bits(0,9,R);
     nc  2.4   , hv  rc5   ;   if bits(20,29,R) = pass no 2 then
c3h: pm  rb8   , ud  s14   ; write pause text:
     sy  64    , sy  58    ;    begin
     sy  62    , sy  0     ;      select(type writer);
                           ;      writecr; write LC;
c4:  cln -6    , ck  -4    ;      write black; write space;
     ga  rb1   , nc  2     ;      write M as 6 characters;
b1:  sy [char] , hv  rc4   ;      comment only one word no CR;
     lyn rb1   , ud  s16   ;      lyn; select(normal);
     can p     , hv  rc7   ;      if p=0 then goto finis;
c5:                        ;    end;
c5h: grn s41e13, ud  s15   ; read start:  sum := 0;
     ca  13[aa], grn rb2   ; read word:   select(secondary reader);
b2:  lyn rb1   , hh  rc5   ;   if code fresh in core then while lyn ≠ <aa> do
                           ;   code fresh in core := f;
     pa  rb3   t   5       ;
c6h: lyn rb4   , tl  -7    ;   R := next word from reader; marks in RC;
b3:  bt[charct]V t -1  NT  ;   if parity error then
     hv  rc9           IZC ;    begin sum ok := transport ok := f; exit end;
     ly  rb1   , hh  rc6   ;
     tl  10    , ud  s16   ;   select(normal);
b4:  pi [marks]t   1020    ;   store[first core] MRC := R;
d13:                       ;
b5:  gr  e16-1 [first] MRC ;   if -,segment transp then
b6:  qq (rb5)  V t   1     ;    begin R := store[first core]; exit end;
     arn(rb5)  , hv  rc9   ;   first core := first core + 1;
     ar  2     D       LRA ;
     ar  1     D       LRB ;   sum := sum + R + RC pos 9;
     ac  s41e13, pp  p-1   ;   p := p-1;
     ncn p     , hhn rc5   ;   if p ≠ 0 then goto read word;

     arn s41e13        IZB ;   sum ok := sum = 0;
     pm  rb7   , arn se13  ;   M := ready text;
     tk  20    , ca 1.0+1.4;   if pass no 1 (e13) ∧ sum = 0 then
     hh  rc3           LZB ;    goto write pause text;
c7:  pm  se13              ; finis:   M := e13;
c8:
c9:  ps [save s],hr  s1    ; exit:  s := save s;  return;

b7:  tready;               ;
b8:  tpasses;              ;
```

```
d1:  [Init GP, paper tape version]

a2:  pp [e4]   , arn p5    ;   GP base := first track translator - 1;
     sr  p3e28 , gr  p11   ;
a:   arn p5    , hs  pe11  ;
     sk  pe28  t   40      ;   GP except fixed GP to first track ff;
     arn p3e28 , ac  p5    ;
     sc  p4    , it  -1    ;   first track := first track + GP tracks;
d23: ncn[tracks],hv  ra    ;   available tracks := available tracks
     vk (p5e11), hh  ra2-37;                     + GP tracks;
     qq [GP sum ]          ;   goto Init comp buffer or paper tape version
e                          ;
d d4 = i-d1                ;   d4 := init GP code length;



a17: pa  r1    t   100     ; spaces:
     bt [100]  t   -1      ;   for i := 1 step 1 until 100 do
     sy  0     , hv  r-1   ;     writechar(0);
     hr  s1                ;

d d2=i, d3=1d2, d5=-d4+184 ;
d d6=d2-39d4-1, d7=39d4    ;
d d8=d2+200,d9=1d2,d11=d2-1;
u b11                      ;
e                          ;
e                          ;
s                          ;
```

[Call: algol, <list> <

<list> ::= <name>|<lineinterval>|s|i|d|n|<empty>|<list><list>

<name> ::= <name of sy-medium>|<name of input medium>

The sy-medium, called normal out, will be used for possible output of source program, pass information and pass output. If no name in the list describes a sy-medium the output unit will be the current selected Help output medium. The translator will be searched unser the name <name>. This is also true if <name> describes ly-medium (transient translator). If no <name> in the list describes a input medium, the translator will be searched unser the name : ga4.

<lineinterval> :== <help number>

Every <lineinterval>th line in the source program will be copied to normal out unit. If no <lineinterval> is present, no output of the source program will be made.

s (skip between punch-off and punch-on)
The parameter will cause program text between punch-off and punch-on to be skipped.

i (information wanted)
Pass information will be output on normal out unit.

d (disc mode)
The disc will be used in a mode which may give fewer head movements during translation of large programs. Experimental facility.

n (no indexcheck)
No check of references to subscripted variables is generated.

Source program and help input medium.

If the help input medium is typewriter without explicitly having been specified as such, then the source program medium will be reader, and after translation the help input medium will again be typewriter.
In all other cases the source input medium will be the current help input medium and after translation the help input medium will be the last used source program medium.

Error output will appear on the current selected help sy-medium.

Alarm output will appear on helps alarm output unit.

Type out: typewriter output.

Type in: typewriter input.]

```
<-d55+1,iversion
s>


[Here follows STOPCODE, CLEARCODE]

b k=d1, i=120, a25, b27            ;
d a=i-3                            ;

[ 3e4] qq  [lineinterval.39]       ; Translator parameters:
[ 4e4] qq  [no of tracks in work.39]; see: A Manual of Gier Algol 4
[ 5e4] qq  [first track of work.39] ; section 13.2.5;
[ 6e4] qq  c64.9-1.9+c63.19        ;
[ 7e4] qq  -1.2+d17.19+1016.29     ;
[ 8e4] qq  1.9+d11.39              ;
[ 9e4] qq  17.29+d17.39            ;
[10e4] qq  1.7                     ;
[11e4] qqf -1.2+63.29+1023.39      ; also used as mask
[12e4] qqf 6.39                    ; also used as constant
[13e4] qqf 6.19+1.39               ; also used as constant
[14e4] qqf d14.39,                 ; also used as constant

b25: pmn 1.3   D X     IZA ; NEXT WORD:   i := 0;  first char := true;
a18: tl  -7    , ly  a19   ; NEXT CHAR:   i := i + 1; RM := RM shift -7;
a19: pi [char] t   508 LZA ;   R := R + lyn pos 9;  if first char then
     hs  c24           LT  ;    begin  RC := bits(8,9,char); first char:=false end;
     qq  a22   X           ;   if char<0 then alarm(‡<parity›);
     hv  a18   X       LZ  ;   if i≠7 then goto NEXT CHAR;
     xr        , tl  3     ;   RM := RM shift 3;
     gr  p183  t  1    MRC ;   c:=c+1; core[183+c] := R mark RC;
[-1] ga  a20   , grn r     ;   if first word then
a20: ncn [words] t  -1     ;    begin words:=RADDR; first word:=false end;
     hv  s1                ;   words:=words-1; if words≠0 then goto NEXT WORD;
a13: grn r2                ; NOT BUFFER:
a12: pm  2a    X t  1  IRC ; BUFFER MEDIUM TRANSLATOR:
     pm  e1    X t  1  LRB ;   move translator parameters
a16: gr [2e4]  t   1       ;   to core[3e4:14e4]; comment in case of
     hv  a12           NRA ;   buffer medium the parameters
     gt  b7    , vk  960   ;   11e4:14e4 are fetched from help;
     vk  d11   , sk  a24   ;   to drum(help parameter track,
     vk  d11   , ps (a16)  ;            AFTER TRANSLATION);
     pp (s170) , pin 0     ; CHECK SUM GP:  R :=0;
b:   ar  s169  t   1   IRC ;   for p:=word size GP step -1 until 0 do
[1]  gi  a20-1 , ar  a20-1 ;    begin
     ck  0     , pp  p-1   ;     R := R + core[184e4+p];
     bs  p     , hv  b     ;     R := R + marks pos 9; clear R00;
     hs  c24           NZ  ;    end; if R≠0 then alarm(‡<sum›);
     qq  a21   , hv  s170  ;   goto first word of GP;

a22: tparity;              ;
a21: tsum;                 ;
```

```
[The following code is placed on Help parameter track during translation
and is entered at b11 with description of translated program in R]

                                ; AFTER TRANSLATION:
a24:  qq [Help param(0)]        ;   This word is used by Help and is set below;

[The instruction in b7 is replaced by :  qq
 if the translator is not on tape]

<d41                            ;   if buffermedia treated
b7:   is  64    , us s[unit];     ∧ tape then rewind translator tape;
>                               ;
b11:  ps  rb1   V  -c41 NZ      ;   if R≠0 then set return(OK) else
[1]   ps  c45-c41               ;   set return(SET TYPEWR);
b24:  gr  rb11  , pm [7e4]      ;   comment SET TYPEWR is an entry in Help;
      gm  r1b11 , vk  960       ;   save description of translated program;
      vk  1d9   , lk  40        ;   save description of latest input medium;
      vk  960   , hv  c53       ;   restore main Help; return to s+c41;
b1:   sy  64    , sy  38[o] ; OK:   writecr; writetext(¢<<ok¢);
      sy  34[k] , hsn c2        ;   get free;  to core(first catalog track);
      lk  d14   , vk (c4)       ;   wait drum;
      arn 3d14  , ac  39d14 ;   checksum := checksum - work area word;
      tl  -32   , tk  32        ;   work are word := bits(0,7,work area word) pos 7
      ar  rb11  , sc  39d14 ;   + description of translated program;
      gr  3d14  , sk  d14       ;   checksum := checksum + work area word;
      vk (c4)   , arn r1b11 ;   modified catalog track back to drum;  wait drum;
b4:   bs  0     , hv  c45       ;   if restore typewriter then goto  SET TYPEWR;
      ps  rb6   V       LT  ; MEDIUM DESCR:  R:= description of latest input med;
      qq  rb8   , ps  rb5       ;   if R<0 then set return(EXIT)
b5:   hh  3c28                  ;   else set return(SKIP);  ENTRY NO GET TRACK;

[The code 3c28 will initialize the input medium described in R
 and return to s+1 in case of ly-medium or drum medium.
 Otherwise return will be made by: hs (s-1). Cell c28 will hold
 number of skipped characters]

[1b5] pm  -2    , hsn c3    ; SKIP:
      lk  d14   , vk (c4)       ;   select track(bits(24,39,core[-2]));
      ps  r-3   , arn c28       ;   to core(place d14);  set return(r-3);
      sr  rb21  , gr  c28       ;   for i:=1 step 1 until skipped characters do
      hv  c27           NT      ;    read internal; comment return by: hr s3;

d b6=i-1                        ; EXIT:  select track(Help parameter track);
      vk  960   , vk  d11       ;   goto RETURN FROM AUX PROGRAM;
      hv  c51                   ; BUFFER INPUT MEDIUM:
<d41                            ;
b8:   arn rb20  , il  0         ;   only loaded if d41 is positive;
      arn c29-1 , ar  d14       ;   core[-6] := -4 pos 9 + 1 pos 19 + 7 pos 39
      gr  -6    , srn d14       ;   +buffer[4];
      sr  rb21  , ud  s-2       ;   core[-5] := -buffer[4] - 1;
      gp  -6    , hv  r1b5      ;   goto SKIP;
×b7: b8:>                       ;
b20:  qq  d14.9+1.19+4.39       ;
b21:  qq  1.39                  ;

<i-194                          ;
b21:>                           ; test load address;
```

```
b10:                          ; ENTRY CALL algol:
<d35-2,us(-31)  t  -96>       ;   if I am on tape then rewind my tape;
      pm  d13   , gm  a24     ;   save param(0); comment used after translation;
      pp  d13-1               ;   p:=base parameterlist;
a1:   gp  a3    , hs  c52     ; SCAN PARAMETERLIST: GET PARAM;
      hv  a5    , hh  a4      ;   if number then goto LINEINTERVAL;
      hv  a7    , tl  -7      ;   if letter then goto SPECIAL BITS;
<d41, nc  -4    , ca  -3      ;   if end list then goto AFTER SCAN;
a2:   qq  e8    , hs  c24     ; TRANSLATOR MEDIUM OR NORMAL OUT:
a3:   ps [p]    , nc  -2      ;   if kind=4  ∨ kind=5  ∨ -,buffermedia treated ∧
× a3: ps [p]    , nc  0       ;     (kind=1  ∨ kind=2  ∨ kind=3) then alarm;
a2:   hs  c24             NT  ;   address of translator medium name :=p;
      qq  e8    , nc  -2      ;   if kind≠2 then normal out := bits(10,19,areaword);
>     gs  a10   , hv  a1      ;
      ck  17    , ga  9a      ;   goto SCAN PARAMETERLIST;
a4:   hv  a1    , ps  -1      ; SPECIAL BITS:   s:= -1;
      ca  18[s] , psn s-1.2   ;   if letter s then s := s - 1 pos 2 else
      ca  57[i] , psn s-1.4   ;   if letter i then s := s - 1 pos 4 else
      ca  52[d] , psn s-1.6   ;   if letter d then s := s - 1 pos 6 else
      ca  37[n] , psn s-1.7   ;   if letter n then s := s - 1 pos 7
      hh  b23             NZ  ;   else alarm;
      pi (10a)  , it  s       ;   bits := (-5 pos 9) mask s;  comment 10e4;
      pi  -5    , hv  a6      ;   goto SCAN PARAMETERLIST;
a5:   gr  3a    , pi (10a)    ; LINEINTERVAL:   lineinterval := R;
      pi  1.5   t  -17  NZ    ;   if lineinterval≠0 then set bit(5,bits);
a6:   gi  10a   , hv  a1      ;   goto SCAN PARAMETERLIST;
a7:   arn 9a    , pi (10a)    ; AFTER SCAN:
      nc  0     , hv  b19     ;   if (lineprint wanted
      gk  9a              LPA ;         ∨ pass information wanted)
      gk  9a              LPB ;      ∧ normal out = 0 then
b19:  gk  a1    , it (a1)     ;   normal out := by;
      pt  9a    , pm  1c26    ;   error out := by;
      tln 6     , tk  3       ;   alarm out := Help alarm out;
      ac  9a    , hsn c2      ;   get free;
a10:  pp  a23   , ps  a15     ;   p := address of translator medium name;
      lk  d14   , vk (c4)     ;   set return(TRANSLATOR MEDIUM);
      pm d14+d45, tl  7       ;   to core(first catalog track);
      tln 16    , gr  4a      ;   set number of tracks and first track
      tln 16    , gr  5a      ;   of work in 4e4 and 5e4;
```

```
     arn -5    , pm  -2    ; INITIAL SOURCE INPUT MEDIUM:
     hh  a8            LA  ;   if bits(40,41,core[-2]) = 0 then
<d41,hv  a9            LB  ; READER OR TYPEWRITER:
>    cln -14               ;    begin core[7e4] := -1 pos 2 + reader pos 19;
     pa  b4    V   1   LO  ;     if core[-2] = vyn [by] t [mask] then
     it (-2)   , pt  7a    ;     part 2 of core[7e4] := part 1 of core [-2]
a8:  hv  c52   , gm  7a    ;     else restore typewriter := true;
                           ;    GET PARAM; comment return by hh s+2;
                           ;   end else if bits(40,41,core[-2]) = 2 then
[1a8]tl  -30   , sr  14a   ; DRUM: begin core[7e4] := core[-2];
     pm (-3)   D X         ;   R := (part 1 of core[-5] )  pos 39;
     ar  a11   , tk  -30   ; ADD: core[7e4] := core[7e4] + ((R - d14)×6
     ml  12a   , tln 55    ;    + 5 + part 1 of core[-3]) pos 23;
     ac  7a    , hv  c52   ;     GET PARAM; comment return by hh s+2 (=a15+2);
<d41,                      ;     end else
a9:  gm  7a    X       IZA ; BUFFER MEDIUM:
     arn 13a   , il  0     ;   begin core[7e4] := core[-2]; M := core[-5];;
     arn 0     , mb  b22   ;    R := buffer[1]; clear bits(3,23,R);
     ac  7a    , srn 3     ;    core[7e4] := core[7e4] + R;
     hv  c52           LZA ;    if core[-5] = 0 then GET PARAM;
     gr  14a   , sr  b21   ;   d14 := -buffer[4];
     gr  -5                ;    core[-5] := -buffer[4] - 1;
     hh  1a8   X           ;    swap; goto ADD;
>                          ;   end;
d a15 = i-2                ;   define return from GET PARAM;
<d41,qq                    ;
     mb  11a   , gr  11a   ; TRANSLATOR MEDIUM:
     ca  0     , it  a13   ;   core[11e4] := bits(0,2,area word) pos 9
     pt (c15)  V   a12 NT  ;   + bits(24,39,area word);
     grn b7    , hv  a17   ;   if kind ≠ 3 then tape := false; comment the
     nc  3.2   , grn b7    ;   instruction in b7 is replaced by: qq;
                           ;   if kind gr 5 then
×    pt (c15)  V   a13 NT  ;    begin tape := false; goto TRANSIENT TRANSLATOR end;
     hv  a17               ;   if kind = 0 then set return(NOT BUFFER) else
     mb  11a   , gr   11a  ;   set return(BUFFER MEDIUM TRANSLATOR);
>    arn (c15) , nc  10    ;   if part 1 of spec word ≠10 (GP tracks)
     nc  11    , hh  b23   ;    ∧ part 1 of spec word ≠11
     ck  10    , gt  b26   ;   then alarm(param);
     grn a20-1 , hs  b26   ;   TEST;
     arn 2c    , ga  r1    ;   R := areaword;
     pi [bits] , pp  d13-1 ;   p := base parameter list;
     hh  1c58          LPA ;   if program bit then PROGRAM CALL;
     qq  e8    , hs  c24   ;   alarm(kind);
```

```
a17: vy  d17   , lyn a19   ; TRANSIENT TRANSLATOR:  first word := true;  c := 0;
     nc  13    , hh  r-1   ;   select(reader);
     pp d14-183, hs  b25   ;   NEXT WORD;
     pa  b26   , hs  b25   ;   R := NEXT WORD;
     arn 2d14  , pm  1d14  ;   M := first word from reader;
     ps  b25-1 , gt  r1    ;   set return(NEXT WORD);
b26: it  -184  , pp [e4]   ; TEST:   p := e4 + (if TRANSIENT TRANSLATOR
                           ;                      then 0 else -184);
     bs  p494  t   503     ;   if p>9 ∧ p<18 then
     gr  p185  V       MC  ;   core[p+185] := R;
b23: qq  e5    , hs  c24   ;   else alarm(param);
     gm  p184          MA  ;   core[p+184] := M;
[2]  it  p7    , pt  b24   ;   save 7e4;
[3]  it  p2    , pa  a16   ;   save 2e4;
     hv  s1                ;   return;


a11: qq  5.9               ;
b12: tga4;                 ;   initial name of translator;
d a23=b12-2                ;
b22: qq  -1.2              ;

<i-280                     ;
b23:>                      ;   test load address;
```

```
i=i+39, d=k-d1                  ;
b k=d42,i=0,a10                 ;
a1=d19-960                      ; a1=group no for image
a=d,<d35,a=1>                   ; a=no of blocks

<d39                            ; if aux only then
i=d2,hs1                        ; begin
     hv a5                      ;  (if aux reserved then
<d36,tres;                      ;
     qqf a.39                   ;   res,no of blocks,
x<d39                           ;    else
     tset;                      ;    set,aux kind,no of blocks,
     qqf d35.39                 ;    type in)
     qqf a.39                   ;
[after i follows STOP,SUM and a sum character]
ia base,algol
s
[STOP,CLEAR]
><d39                           ;
   qq  39,                      ; concat pid 0,
   qq  57,                      ;
   qq  52,                      ;
   qqf                          ;
   talgol;                      ; algol,spec<
   qqf d.9+b10.19+120.29        ;
   qqf,                         ;

a5: hs 1                        ;
    hv a6                       ;
    tmove;                      ; move,b loadplace,algol<
    qq 50,                      ;
    qqf d.23+d1.39+a1.29-a1.33  ;
    talgol;                     ;
    qqf,                        ;

a6: hs 1                        ;
    hv a7                       ;
    tsetsum;                    ;
    talgol;                     ;
    qqf,                        ;
d2=i                            ; end else
a7: hsf 2                       ;
x                               ;
i=d48                           ;
   qq d35.2+11.7+d36.5+d.23+d1.39,; load to primitive catalog;
   qq,                           ;
   talgol;                       ;
   qqd.9+b10.19+120.29           ;
d48=i,qqf                        ;
>                                ;
d1=d+d1   ;
e         ; end image load
e         ; end algol
[after i follows STOP,SUM and a sumcharacter]
iaalgol
s
```

```
<-d55+1, iversion
s
>
[Here follows STOPCODE, CLEARCODE]

b k=d1, i=40d13, a15, b15  ;   begin binin
b=79i                      ;   define base of working locations
     pp  d13-1 , hs  c52   ; ENTRY TO BININ: get first parameter;
     hh  r1    , hh  r-1   ;   if single then get next; if end list then alarm;
     hv  c58   , ga  rb11  ;   indicator:= bits 0-9; R:= area word;
b11: pi  0 [area bits]     ;
     qq        V       LQA ;   if work area then R:= work as output;
     arn d45+d14       LTB ;
     tl  -32   , tk  25    ;   if kind ≠ drum then alarmprint(‹<kind›);
     nc  0     , hv  c57   ;   now track read is false (ZB = 0)
     tln 16    , gr  r3b   ;   upper:= first track + no of tracks;
     tln 16    , ac  r3b   ;
     gr  r2b   , gr  r5b   ;   track:= destination base:= first:= first track;
     gr  r1b   , ly  rb1   ;
     nc  13    , hh  r-1   ;   for i:= lyn while i ≠ aa do ;

a:   pmn r2b   , hs  c3    ; start track: select track(track);
     pp  0     , lk (rb4)  ;   read track to store buffer[0];
b6h: vk  960   , pp  0     ;
     [trackrel]            ;   p:= track rel - 1;
a1:  pp  p-1   , psn ra2   ; read label: prepare return to label;
a4:  ly  rb1               ; procedure inchar; Radr:= words:= lyn;
     ac (rb7)  DV      NT  ;   sum:= sum + Radr;
a6:  qq  rb8   , hs  c24   ;   if Radr < 0 then alarmprint(‹<parity›);
     qq (rb9)  t  1        ;   characters:= characters + 1; return;
a2:  hh  s     , ps  ra3   ; label: prepare return to next word;
     bs (rb1)  Xt   63     ;   if words ≥ 64 then goto special;
a3:  hv  ra10  , it  -1    ; next word:
b1:  can -1    , hh  ra1   ;   if words = 0 then goto read label;
     [words]
     pmn 1.3   DX      IZA ;   for i:= 1 step 1 until 6 do

a5:  tl  -7    , ly  r1    ;   begin pack; inchar;
     pi  -1    t 256   LZA ;     if i = 1 then RC:= R8 - 9;  ZA := 0;
b7:  ac  0     DVX     NT  ;   sum := sum + char;
     [sum]                 ;
     hv  ra6               ;     if Radr < 0 then alarmprint(‹<parity›);
     hv  ra5   X       LZ  ;   end;
b9:  qq  0     Xt6         ;   characters:= characters + 6
     [characters]          ;
a7h: tl  3     , pp  p1    ;   R:= packed word;
b4:  gr  p130d13 X     MRC ; outword: p:= p + 1; storebuffer[p]:= R, RC;
     hk  ra9           NZB ;   if not busy ∧ -, trackread then readtrack;
     bs  p472  , hh  s     ;   if p < 40 then return;
```

```
a8:  hs  ra9          NZB ; procedure writetrack; if -, trackread then read track;
[1a8]arn r2b   , sr  r3b  ; writetrack no read:
     hv  c61          NT  ;   if track ≥ upper then alarmprint(ɬ<full>);
     pm  r2b              ;   track read:= false;
     hsn c3           IOB ;   select track(track);
     arn rb    , ac  r2b  ;   track := track+1;
     arn(rb4)  , pp  0    ;   R:= last word stored; p:= 0;
     sk (rb4)  , pm  rb4  ;   write track from storebuffer[0];
     it (rb5)  , pa  rb4  ;   exchange (store buffer, drum buffer);
     ud  rb4              ;   storebuffer[0]:= R, RC;
     ga  rb5   , hh  s    ;   Raddr ≠0; return;

a9:  arn r2b   , ar  rb   ; procedure readtrack;
     sr  r3b              ;   if track + 1 ≥ upper then return;
     hr  s1           NT  ;
     ar  r3b   X          ;   select track (track + 1);
     hsn c3           IZB ;   read track to drumbuffer[0]; trackread:= true;
b5:  lk  171d13 , hr s1   ;   return;

a10: it (rb9)  , pt  rb2  ;   characters 1:= characters;
     it (rb7)  , pt  rb3  ;   sum 1:= sum;
     hsn ra4   , tk  -7   ;
     hs  ra4   , tk  -7   ;   k:= (inchar shift -14)
     hs  ra4   , xr       ;   + (inchar shift -7) + inchar;
     ca  64    , hv  ra11 ;   if words = 64 then goto repeat;
     ca  65    , hv  ra12 ;   if words = 65 then goto new destination;
b2h: tl  43    , ca  -1   ; check sum:
b3h: ck  10    , nc  -1   ;   if characters 1 ≠ k : 1024  ∨ sum 1 ≠ k mod 1024
     qq  rb10  , hs  c24  ;   then alarmprint(ɬ<tapesum>);
     hs  r1a8  , hs  c2   ;   write track no read; get free;
     arn 2b    , pm  5b   ;   R:= track; m:= destination base;
     pi (rb11) , hv  c74  ;   indicator:= area bits; goto adjust special;

a12: tl  43    , gt  rb6  ; new destination: trackrel:= k mod 1024;
     tl  -30   , ar  r1b  ;
     gr  r4b   , gr  r5b  ;   work:= destination base:= first + k : 1024;
     hs  r1a8  , pm  r4b  ;   write track no read; track:= work;
     gm  r2b   , hv  ra   ;   goto start track;

a11: tln 23    , ps  r1   ; repeat:
     hv  r1    , arn r4b  ;   for work:= k step -1 until 1 do
     sr  rb    , gr  r4b  ;
     hh  ra1          LT  ;   outword(last word);
     arn(rb4)  , hh  ra7  ;   goto read label;

b8:  tparity;              ; alarm messages.
b10: ttapesum;             ;

< i-b, i track, binin
> i=b


     qq  1.39              ; count
[1b: first.39              ; first track of area
 2b: track.39              ; output buffer will go to this track when full
 3b: upper.39              ; first track after area
 4b: work
 5b: destination base      ; first track changed after new destination]
e                          ; end binin code
```

```
i=i+39, d=k-d1                  ;
b k=d42,i=0,a10                 ;
a1=d19-960                      ; a1=group no for image
a=d,<d35,a=1>                   ; a=no of blocks

<d39                            ; if aux only then
i=d2,hs1                        ; begin
     hv a4                      ;  (if aux reserved then
<d36,tres;                      ;
     qqf a.39                   ;   res,no of blocks,
x<d39                           ;    else
     tset;                      ;    set,aux kind,no of blocks,
     qqf d35.39                 ;    type in)
     qqf a.39                   ;
[after i follows STOP,SUM and a sum character]
ia base,binin
s
[STOP,CLEAR]
><d39                           ;
   qq  39,                      ; concat pid 0,
   qq  57,                      ;
   qq  52,                      ;
   qqf                          ;
   tbinin;                      ; binin<
   qqf,                         ;

a4: hs 1                        ;
    hv a5                       ;
    tmove;                      ; move,b loadplace,binin<
    qq 50,                      ;
    qqf d.23+d1.39+a1.29-a1.33  ;
    tbinin;                     ;
    qqf,                        ;

a5: hs 1                        ;
    hv a6                       ;
    tsetsum;                    ;
    tbinin;                     ;
    qqf,                        ;
d2=i                            ; end else
a6: hsf 2                       ;
x                               ;
i=d48                           ;
   qq d35.2+11.7+d36.5+d.23+d1.39,; load to primitive catalog;
   qq,                           ;
   tbinin;                       ;
d48=i,qqf                        ;
>                                ;
d1=d+d1   ;
e         ; end image load
e         ; end binin
[after i follows STOP,SUM and a sumcharacter]
ia binin
s
```

[here follows STOPCODE, CLEARCODE]

```
b k=d1, i=40d13, a25, b15  ; begin binout
b a10                      ; begin outparam

     can(2c22) , vy 32     ; ENTRY OUTPARAM: if no output unit then select(p);
<d35-2, us(-31)t   -96>    ;   if aux kind = tape then rewind tape;
     sy  58    , sy  64    ;   writecr;
a:   pmn d13   t   1       ; next: param:= param + 1;
     hv  a8            LC  ;   if list[param] = end then goto end list;
     hv  a4    X       LA  ;   if list[param] -< letter then goto letter;
     hh  a6    X       LB  ;   if list[param] -< number then goto number;
a1h: arn a2    , cl  -6    ; text:
     tk  -4    , ga  a3    ;   char:= next char (list[param]);
     ca  15    , hv  a     ;   if char = end word then goto next;
a2:  ca  10            IB  ;   if char = end text then
     [1.39]                ;   begin write comma; goto next
     sy  27    , hv  a     ;   end;
     ca  63    , it  1     ;   if char = 63 then char:= CR;
a3:  sy  0     , hhn a1    ;   writechar(char); goto text;

a4:  sy  14    , ga  r1    ; letter: write underlining;
     sy  0     , hv  a     ;   writechar (list[param]); goto next;

a5:  ck  10    , gr  (a)   ; number: simple print(list[param]);
a6h: sy  59    , hs  d28   ;   for i:= 1 step 1 until 3 do
a7:  bt  3     t   -1      ;   begin list [param]:= list[param] shift 10;
     arn(a)    , hv  a5    ;     writepoint;
     pa  a7    t   3       ;     simple print (list[param]);
     sy  27    , hv  a     ;   end; writecomma; goto next;

a8:  sy  17    , hv  -9    ; end list: writetext(<); return to help;

e                          ; end outparam

d=d19-960                  ;
b:   qq d.29-d.33+d16.39   ; base: initially image base
[1b] qq 1023.9+1023.29     ; mask
[2b] qq 1023.19+1023.39    ; mask
[3b] qq 26.39              ; image length
[4b] qq 510.39-1c54.39     ; max length of bin 0 tape
[5b] qq 1.39               ; 1
b12: qq 40.39              ; length tabel: drum
[1b12]qq d53.39            ; disc
[2b12]qq 512.39            ; carr
[3b12]qq 400.39            ; tape

a=240d13, b8=6a            ; define outbuf base
[a:  length
1a:  label
2a:  previous word
3a:  alike
4a:  blocks
5a:  work, skipped words]
```

```
a18: pm  3b    , gm  4a    ; ENTRY BINOUT: blocks := 26;
<d35-2,us(-31) t -96>      ;   if aux kind = tape then rewind tape;
     can(2c22) , vy  32    ;   if no of output selected then select(p);
b1:  pp  d13-1 , hs  c52   ; NEXT PARAM: get next param;
     hv  c58   , hv  a1    ;   if number then param alarm;
                           ;   if single then goto domain;
a21h:hv  a2    , gr  b     ;   if end list then goto write end label;
     hv  c57           LT  ;   base:= area word; if kind < 0 then kind alarm;
     tl  -7    , ga  b11   ;   store kind;
     tl  -25   , tln 16    ;   blocks:= bits 8-23;
     gr  4a    , hh  b1    ;   goto next param;

a2:  ncn(b6)   , hv  -9    ; WRITE END LABEL: if only bin 0 then return to help;
     sy  66    , it (b7)   ;   write end mark;
     pa  -1    , arn -1    ;   out label (characters × 1024 + sum);
     ck  10    , hs  a5    ;
     hv  -9                ;   return to help;
a1:
b11: ps [kind] , pm  p2    ; DOMAIN: interval:= next word in param list;
     ca  16    , pin 16    ;   if single = 0 then bin 0:= PB:= true;
     ca  37    , pin 0     ;   if single = n then bin 0:= PB:= false;
     qq        V       LC  ;   if next word in param list ≠ number then
     hv  a19           LB  ;   begin
     hv  c58           NZ  ;     if single ≠ 0 ∧ single ≠ n then param alarm;
     pmn 4a    , mln sb12  ;     length:= blocks × length[kind]; first word:= 0;
     hhn a20   X           ;   end else
a19: pp  p1    , ca  50    ;   begin prepare skip of next param;
     hh  a21   X           ;     if single = b then goto set base;
     hv  c58           NZ  ;     if single ≠ 0 ∧ single ≠ n then param alarm;
     xr        , mb  1b    ;
     cl  30    , arn p1    ;     M:= first word:=
     mb  2b    , ml  sb12  ;     bits 0: 9 (interval) × 40 + bits 10:19 (interval);
     cln -20   , ck  -20   ;     R:= last word:=
     gm  5a    , sr  5a    ;     bits 20:29 (interval) × 40 + bits 30:39 (interval);
a20h:ar  5b    , gr  a     ;     length:= last word - first word + 1;
     grn c81   , gm  1a    ;   end; set no label check; label:= first word;
     dln sb12  , gm  5a    ;   skipped words:= first word mod length[kind];
     vk  960   , vk  c63   ;   select and read init medium 1;
     lk  c28   , ar  b     ;   R:= first word : length[kind] + base;
     qq  40c28 , vk  960   ;
     pa  3c28  , hs  2c28  ;   medium:= false; init medium;
     pmn 1c            IPA ;   PA:= medium = drum;
     gp  b1    , hs  c3    ;   save param address; select first block
     lk  d14   , vkn 960   ;   and read it to text buffer; prepare no fault;
     hv  c70           NZ  ;   for i:= skipped words step -1 until 1 do
     ps  r-2   , arn 5a    ;   begin if fault then fault alarm;
     sr  5b    , gr  5a    ;     call(get next word);
     hv  a7            NT  ;   end;
     pa  b3    t   -1      ;   previous marks:= not existing;
     pp  0     , grn 3a    ;   out:= 0; alike:= 0;
```

```
[Output prelude]
b5:  hv  a17          LPB ;   if bin 0 then goto output bootstrap;
     pa  b5    t   c58     ;   next bin 0 should give param alarm;
b6:  can 1     , hv  a4    ;   if -, bin normal printed then
     bt  100    t   -1     ;   begin
     sy  0     , hv  r-1   ;     outsp(100);
     sy  13    , pt  -1    ;     write aa; sum:= 0;
     pa  b7    t   1       ;     characters:= 1; [prepares final 66-mark]
a4:  sy  65    , pmn 1a    ;   end; write labelmark;
     dl  b12   , gr  1a    ;   outlabel(label : 40 × 1024 + label mod 40);
     cln -10   , ar  1a    ;
     pa  b6    , hs  a5    ;   bin normal printed:= true;

a12: hs  a7                 ; PROCESS NEXT WORD: get next word;
     hv  c70          NZ  ;   if fault then fault alarm;
     qq        X      IRC ;   RC:= marks of word;
     hv  a6    X      LPB ;   if bin 0 then goto out bin 0;
     sr  2a    , gi  1a    ;   if current word = previous word
     hh  a8           NZ  ;
     arn 1a    , ps  a9    ;   ∧ indicator = previous marks then
b3:  nc  -1    , hh  a8    ;   begin
     [previous marks]       ;     alike := alike + 1;
     arn 5b    , ac  3a    ;     output buffer
a8h: hv  a10   , hs  a11   ;   end else
     bs  p-62  , hs  a10   ;   begin outrepeat; if out > 62 then
     pm  c-1   , pp  p1    ;     output buffer; out:= out + 1;
     gm  pb8         MRC ;     outbuf[out]:= previous word:= current word;
     gm  2a    , gi  b3    ;     previous marks := indicator end;
a9=i-1 [Return from outbin 0, output buffer]
     arn a     , sr  5b    ; CHECK LENGTH: length:= length - 1;
     gr  a     , ps  b1-1  ;   prepare return to next param;
     hv  a12         NZ  ;   if length > 0 then goto process next word;
[End of domain]             ;
     hv  a22         NPB ;   if bin 0 then
     sy  64    , it (b7)   ;   begin writecr to stop primitive input;
     pa  -1    , arn -1    ;     out label(characters × 1024 + sum);
     ck  10    , hv  a5    ;     goto net param
a22: hs  a10                ;   end; output buffer;

a11: arn 3a    , sc  3a    ; OUTREPEAT: R:= alike; alike:= 0;
     hr  s1          LZ  ;   if R = 0 then return;
     sy  64    , ck  -10   ;   write repeat mark; R:= R shift -10;
a5:  ga  r1    , ck  -7    ; OUTLABEL:
     sy  -1    , ga  r1    ;   writechar(bits 3:9(R));
     sy  -1    , ck  -7    ;   writechar(bits 36:2(R));
     ga  r2                ;   writechar(bits 29:35(R));
     qq (b7)   t   4       ;   characters:= characters + 4;
     sy  -1    , hr  s1    ;   return;
```

```
a10: can p     , hr  s1    ; OUTPUT BUFFER: if out = 0 then return;
     qq (b7)   t   1       ;   characters:= characters + 1;
     sy  p     , gp  b9    ;   writechar(out); start:= 1;
a14h:pp  1     , pmn pb8   ;   for i:= start step 1 until out + start - 1 do
     ar  1.1   D       LA  ;   begin
     ar  1.2   D       LB  ;     R:= outbuf[i];
b7:  qq  1     Xt   6      ;     M:= marks.2;
a13h:cl  32    , ga  r1    ;     characters:= characters + 6;
     sy  -1    , it  -100  ;     for j:= 1 step 1 until 6 do
     bt  0     , hv  r2    ;     writechar(char j of RM);
     cl  -7    , hh  a13   ;
     pa  r-2   , pp  p1    ;
b9:  can -1    t   -1      ; OUT SEGMENT:
     pp  0     , hr  s1    ;   end;
a15: hh  a14               ;   out:= 0; return;

a7:  hv  c71           NPA ; GET NEXT WORD: if -, drum medium then
     arn(c-2)  Dt  1       ;   goto get word; word address:= word address + 1;
     nc  40d14 , hh  a23   ;   if word address = top of buffer then
     hs  c16           M   ;   begin select next track;
     lk (c-2)  t   -40     ;     word address:= first in text buffer;
a23h:vk  960   , pmn(c-2)  ;     read track end;
     gm  c-1   , hr  s1    ;   M:= current word:= text buf[word address];
                           ;   return with no fault;

[The Bootstrap program on track 0 works like this:
      original        2 instr. read   2 + 6 instr. read
s-1   tl -6, ca 0
s     ly r4, hs r-1
s+1   gm s3 Mt -1     gm s7 Mt-1      bs (r4)  IO
s+2                   hv s            tl 12 V IK
s+3                                   hv [513-real length|MR
s+4                                   pi 0 t -49
s+5                                   gr [510-real length|MPCt1
s+6                                   hv  s  IKC
s+7   sk dumped. This word must not be destroyed]

[Bootstrap program, 6 7-bit characters / word.      Bits of: Curr instr, prec.]
b10:   qq   2.6+ 0.13+ 0.20+  0.27+ 0.34+ 0.39  ;hv s             25-39, length
               [bits 29-34, 36-38 is length]
       qq   2.6+ 4.13+ 0.20+  0.27+ 0.34+ 7.39  ; gm s7 t-1 M     25-39, 0-24
       qqf 28.6+ 2.13+ 0.20+127.27+63.34+11.39  ; hv s     IKC    19-39, 0-24
       qq  32.6+ 6.13+56.20+ 64.27+ 0.34+ 0.39  ; gr xx t1 MPC    25-39, 0-18
[4b10] qq   0.6+ 0.13+ 0.20+  0.27+64.34+10.39, ; pi 0 t -49      31-39, 0-24
               [bits 15-20, 22-24 is 510 - length]
       qq  28.6+ 0.13+ 7.20+103.27+49.34+ 8.39  ; hv xx    MR     19-39, 0-30
[6b10] qq   0.6+32.13+96.20+  0.27+ 0.34+ 0.39  ; tl 12 V  IK     25-39, 0-18
               [bits 22-27, 29-31 is 513 - length]
       qqf 24.6+44.13+ 1.20+ 32.27+64.34+ 2.39  ; bs (r4)  IO     19-39, 0-24
       qq   0.6+ 0.13+32.20+ 64.27+ 0.34+ 0.39  ;                  fill , 0-18
       qq             64.20+  0.27+ 0.34+ 0.39  ;                  fill
```

```
a17: arn a     , sr  4b    ; OUTPUT BOOTSTRAP:
     hv  c58           NT  ;   if length > max length then param alarm;
     sy  14    , sy  17    ;   writetext(↟< ≤ ↠);
     pmn a     , tl  36    ;
     tk  5     , ac  b10   ;   pack length into boot[0];
     tln 4     , ac  b10   ;
a3:  arn 507   Dt   3      ;
     tk  -30   , sr  a     ;   pack 510 - length into boot[4];
     tl  -3    , tk  19    ;
     ac  4b10  , tln 18    ;
     ac  4b10  , ud  a3    ;
     tk  -30   , sr  a     ;   pack 513 - length into boot[6];
     tl  -3    , tk  12    ;
     ac  6b10  , tln 11    ;
     ac  6b10  , pp  b10-b8;   p:= start:= first boot - out buf base;
     pa  b9    t   10      ;   out:= 10;
     pa  b7    t   1       ;   characters:= 1; [prepare final 64-mark]
     pt  -1                ;   sum:= 0;
     pa  b5    t   c58     ;   next bin 0 should give param alarm;
     ps  a12-1 , hv  a15   ;   call(out segment); goto process next word;

a6:  ar  1.1   D       LRA ; OUT BIN0:
     ar  1.2   D       LRB ;
     qq (b7)   t   7       ;   characters:= characters + 7;
     ck  3     X           ;   R:= current word; M:= marks.39;
     cl  34    , ps  -6    ;
a16: ck  -4    , ga  r2    ;   for s:= -6 step 1 until 0 do
     bs  s1    , it  64    ;   writechar((if s = 0 then 64 else 0)
     sy  -1    , ps  s1    ;   + char s of RM);
     bs  s     , hv  1a9   ;   goto check length;
     cln -6    , hv  a16   ;
```

```
i=i+39, d=k-d1              ; d = no of tracks
b k=d42, i=0, a10           ;
a1=d19-960                  ; a1 = group no for image
a2=40d13                    ;
a3=a18                      ;
a=d, <d35, a=1>             ; a = no of blocks

< d39                       ; if aux only then
i=d2, hs 1                  ; begin
     hv  a4                 ;
< d36, tres;                ;   (if aux reserved then
     qqf a.39               ;   res, no of blocks,
× < d39                     ;   else
     tset;                  ;   set, aux kind, no of blocks,
     qqf d35.39             ;   typein)
     qqf   a.39             ;

[after i follows STOPCODE, SUMCODE and a sum character]
ia base, binout
s
[STOP, CLEAR]
>< d39                      ;
     qq 39,                 ;   concat p i d 0,
     qq 57,                 ;
     qq 52,                 ;
     qqf                    ;
     tbinout;               ;   binout, spec, outparam <
     qqf d.9+a3.19+a2.29    ;
     toutparam;             ;
     qqf,                   ;

a4:  hs 1                   ;
     hv  a5                 ;
     tmove;                 ;   move, b load place, binout <
     qq  50,                ;
     qqf d.23+d1.39+a1.29-a1.33     ;
     tbinout;               ;
     qqf,                   ;

a5:  hs  1                  ;
     hv  a6                 ;
     tsetsum;               ;   setsum, binout <
     tbinout;               ;
     qqf,                   ;
d2=i                        ; end else
a6:  hsf  2                 ;
×                           ;
i=d48                       ;
     qq d35.2+11.7+d36.5+d.23+d1.39, ; load to primitive catalog;
     qq,                    ;
     tbinout;               ;
     qq d.9+a3.19+a2.29     ;
     toutparam;             ;
d48=i, qqf                  ;
>                           ;
d1=d+d1                     ;
e                           ; end image load
e                           ; end of binout

[after i follows STOPCODE, SUMCODE and a sum character]
ia binout
s
```

[STOP, CLEAR]

```
b k=d1, i=40d13 + 327, a63, b19 ;
                                ;
```

[Variables and buffer areas:

```
b-4: sizes or save spec
b-3: save free
b-2: new free
b-1: out track
b=44d13
b-19b: entry buf
40b-79b: out buf
80b-322b: used by sum and move
323b =40d13+327: first word loaded.]

b=i-323
a59:
a: [first word of sum and move loaded here]
```

[sum and move]

[A subroutine used for summing and, if the area is reserved, moving
an area given by the two first words of a catalog entry.

The subroutine uses 40 + n times 40 + 3 working locations in front of
it self, where 1 ≤ n, as follows:

b1=i-3-n times 40 - 40;
  b1-19b1: used for first track of init medium and for:
    b1: number of blocks.
    1b1-5b1: core part of medium description
    b3=b1+c82-c28
    b3-5b3: buffer part of medium description
    6b3: words to sum
    7b3: to core word, transport word for il 0
    8b3: working location
  b4=40b1, b=b4+b2 where b2 = n times 40. b2 is originally 200
    b4 ff are used for second track of init medium and
    afterwards as core buffer for summing and for moving of tracks.
  b: new free
  1b: to free word, parameter for us unit if free kind is disc
  2b: area sum

call:
  M:= first free. Will be updated if the area is reserved.
  p:= core address of area word of catalog entry.
  LTA: namelist. Areas of kind 2 or 3 will only be summed if LTA.
  LRA: move it. If the area is reserved it will be moved to first free.
  LRB: sum it. If the area has a sumbit and a sumword the
       sum will be generated. Areas of kind 2 or 3 only if LTA however.
  The address part of first word of the subroutine must hold the
       address of selected track (M). Originally set to point at
       first word after the routine
  hs first word

Normal return:
  hr s1 with:
  R = sum, only relevant if LRB
  M = updated first free
  LTB - LQB = bits(3, 7, areaword). If special area then NPB
  LRA: has been moved.
  LRB: has been summed.

Error reactions: fault alarm

Notes: s and p have been used]

```
b b15, a21                ; Core block sum and move;

b2=200[n times 40]        ; b2 = core buf size
b=i-3, b4=b-b2, b1=b4-40  ; b4 = core buf
b3=b1+c82-c28             ;

a1:  qq  a21   , hv  a2   ; sum and move:
a:   qq  1.39; 1
[1]  qq b2.39; core buf size
[2]  qq 8b3t1; check il word

a2:  arn p     , ga  b10  ;   R:= area;
b10: pi[kindbits]t  899   ;   indicator 3 to 7:= kind bits;
     pi 0.5+0.9 t 1006 LTB ;   if special  ∨ kind > 3 then
     pi 0.5+0.9 t 1006 LT  ;     reserved := sumit:= f
     gm  b     , gm  1b    ;   new free:= to free word:= M
     tl  -32   , tln 16    ;   if reserved then
     ac  b     V       LPB ;     new free:= new free + bits (8, 23, R)
     pi  0.8   t  1021     ;   else moveit:= f;
     tln 16    , sr  1b    ;   if to free word = bits (24, 39, R) then
     pi  0.8   t  1021 LZ  ;     move it:= f;
     pi  0.9   t  1022 NQB ;   if -, sum then sum it:= f;
     bs (b10)  t  1.1-1.9  ;   if bits (0, 2) = 2  ∨ bits (0, 2, R) = 3 then
     pi  0.9   t  1022 NTA ;     begin if -, namelist then sumit:= f; end;
     arn p1    , gs  b13   ;   save s:= s; area sum:= 0
     pin 0.9   t  1022 NA  ;   if no secondary word then sumit:= f
     gr  2b    , vk  960   ;   else area sum:= secondary word;
     hv  a18           NRC ;  if -, (sumit  ∨ move it) then goto exit;

     vk  c63   , lk  b1    ; init medium:
     vk  1c63              ;   get first medium track;
     pt  2b1   t   3b1     ;   medium:= false;
     qq  40b1  , pa  3b1   ;   core part:= 3b1;
     arn p     , hs  b1    ;   init medium (area);
     hs  c24           NZ  ;   if R ≠ 0 then label alarm;
     qq  e9    , pm  5b1   ;   if buffer medium then goto kind 1 to 3;
     hh  a10           LB  ;
a3:  grn 6b3   , pp  0     ; kind 0: words to sum:= 0; p:= 0
a4:  pm  5b1   , hs (a1)   ; next track: to core (curr track, core buf + p);
     lk  pb4   , arn a     ;   curr track:= curr track + 1;
     ac  5b1   , sc  b1    ;   blocks:= blocks - 1; p:= p + 40;
     pp  p40   , gp  b11   ;   buf size:= p;
     arn b1    , bs p-b2+512;   if p < core buf size ∧ blocks ≠ 0 then
     hv  a4            NZ  ;     goto next track;
     hv  a6            NRA ;   if move it then
a5h: pp  0     , pp  p40   ;     begin p:= 0;
     pm  1b    , hs (a1)   ; write next: to drum(to free word core buf + p);
     sk  pb4-40, arn a     ;     p:= p + 40; to free word:= to free word + 1;
     ac  1b    , it  p     ;     if p < bufsize then goto write next
b11: bs[buf size], hh a5   ;     end;
a6:  vk  960   , hsn a14   ;   wait drum;
     hv  a3                ;   sum p words goto kind 0;
```

```
<-d41+1 [if -,buffer media];
a10=c57, a13=c57          ; kind 1 to 3: error exit;
×    [else]               ;
a19: arn 1b    , us (4b3) ; write block (repeat):
                          ;   us(check unit, to free word)
a20: arn 2a    , il (4b3) ; check transp (repeat):
     il  0     , arn 8b3  ;   il (check unit, status to core);
b14: can[check ct]Vt 256 LT;   il (0, status to core); check ct:= check ct + 1;
     pa  b14   , hr  s1   ;   if ok then begin check ct := 0; return) end;
     arn r     , hv  c70  ;   if check count = 4 then faultalarm;
a10h:hr  s-1   , arn 1b   ;   return to (repeat);

     tk  18    , gr  1b   ; kind 1 to 3: to free word:= to free word pos 21 +
     arn 2b3   , ga  1b   ;   bits (0, 9, current block) pos 9 +
     tl  -12   , tln 12   ;   bits (28, 39, current block);

a11: ac  1b    , arn a    ; next block: no of blocks:= no of blocks - 1;
     sc  b1    , arn 2b1  ;   if left in buf = 0 then
     hv  a12          LT  ;     begin
     arn 1b3   , ac  2b3  ;     current block:= current block + block increment
     arn 2b3   , ud  4b3  ;     rep il: il (read block, current block);
     hs  a20   , qq  a11-1;     check transp (rep il); set return next block
     srn 3b3   , gr  2b1  ;     rep us: left in buf:= -block length;
     hs  a19          LRA ;     if move it then write block (rep us);
     arn 2b3   , tl  -12  ;     to core word:= bits (28, 39, current block)
     tln 12    , gr  7b3  ;       + core buf pos 9;
     pa  7b3   t   b4     ;     end;

a12: pm  3b3   , dln 5b3  ;   words to sum:= block length : area word increment;
     gr  6b3   , ac  2b1  ;   left in buf:= left in buf + words to sum;
     arn 1b3   , ac  1b   ;   to free word:= to free word + block increment;
     hv  a16          NRB ;   if sum it then
                          ; sum buf:
a13: arn 6b3   , sr  1a   ;     begin
     qqn              NT  ;     R:= if words to sum < core buf size
     ar  1a    , gr  8b3  ;     then words to sum else core buf size;
     ck  20    , gt  b12  ;     p:= R;
     ar  7b3   , il  0    ;     il(0, to core word + R pos 19);
     arn 8b3   , ac  7b3  ;     to core word:= to core word + R;
b12: sc  6b3   , ppn[tosum];     words to sum:= words to sum - R
> [end buffer media]      ;     end;
a14: hv  a16          NRB ; sum p words:
a15: pp  p-1   , ar  pb4   ;   if sum it then
     ar  2     D      LA  ;     begin
     ar  1     D      LB  ;     for p:= p - 1 while 0 ≤ p do
     bs  p     , hv  a15  ;       area sum:= area sum +
     ac  2b    , srn 6b3  ;         word[core buf + p] + the marks pos 9;
     hv  a13          LT  ;     if 0 < words to sum then goto sum buf
a16: arn b1              ;     end;
     hvn s1           NZ  ;   if no of blocks ≠ 0 then goto return;

a18: arn 2b    , pm  b    ; exit: R:= area sum; M:= new free;
b13: ps[save s], hr  s1   ;   s:= save s; go right back;

a21: [here starts select track]
e    [core block sum and move];
```

```
a1:  dln a9    , ck  -10   ; select track:
     ga  b1    , cln -10   ;   group:= M : 960 + 960; select (group);
b1:  vk [group]t   960     ;   track:= M mod 960; select (track);
     ga  b2                ;   return;
b2:  vk [track], hr  s1    ;


a2:  pp  p1    , arn r     ; get next word: p:= p + 1; fount:= t;
     hs  c15           IZA ;   nextword;
     gm  pb-1          MPC ;   word [p+entry buf-1] MPC:= M;
     hr  s1    X       LZ  ;   if R = 0 then swap return;
a3:  ca  1                 ; caterr: if Raddr = 1 then
     qq  e7    , hs  c24   ;   alarm print (ǂ<catalogǂ);
     hs  a40   , qq  sa5   ;   alarm(ǂ<nameǂ, 1);

a60: pi1.2+1.9 , hh  a61   ; check: start it (0, f, f, t);
a61: pi1.2     , hs  a4    ; list:  start it (0, f, f, f);
a62: pi1.2+5.9 , hs  a4    ; set sum: start it (1, t, f, t);
a63: pi1.2+2.9 , hs  a4    ; compress: start it (2, f, t, f);
                           ; start it(act, name list, force sum, move it, sum it);
a4:  gs  b3    , hsn a6    ;   comment namelist = LTA. force sum = LQB.
     arn(b3)   Vt  a51 NC  ;   move it = LRA. sum it = LRB; R:= 0;
b3:  arn[act]  t   a50 ITA ;   param; if last param then
     ga  b9    , gt  b4    ;     begin act:= act + 3; namelist:= f end;
     tk  20    , ga  b7    ;   R:= action table[act]; main action:= part 1(R);
b4h: gi  b8    , hv [start];   start:= part 2(R); finis:= part 3(R);
                           ;   main in:= indicator; goto start;
a5:  tname;                ;

a6:
b5:  pp d13-1[paramaddr]t1 ; param: p:= param addr:= param addr + 1;
a7:  arn p1            LZ  ; test param: if R = 0 then R:= param[paramaddr];
     hv  s1            LC  ;   if last param then goback
     hh  s1            NC  ;   if name param then go right back;
     bs  p     t  d13  NB  ;   if not first param  ∨ help number then
a8:  qq  e5    , hs  c24   ;   alarm print(ǂ<paramǂ);
     pp  2     , ca 41 [r] ;   print kind:= if Raddr = r then 2 else
     can(2c22) , vy  32    ;     if Raddr = n then 1 else if Raddr = a then 3
     ca  37[n] , ppn 1     ;     else 0;
     ca  49[a] , ppn 3     ;   if print kind ≠ 0 then R:= 0;
     gp  a55   , hv  a6    ;   if print kind = 2 ∧ no output then select(32);
                           ;   goto param;
a9:  m 960                 ;
```

```
a10: srn 3[+rel cat] D 1   ; start compr: rel cat:= 1;
[1]  gr  79b           MC  ;   word [outbuf + 39] MC:= - rel cat - 3;
     gp  -1            MA  ;   half inhibition;
     vy  -1    t   -1d18   ;   by inhibit;
     pm  c12   , gm  b-1   ;   outtrack:= first catalog track;
a11: hs  c2                ; start no name: get free;
     hv  a3            NZ  ;   if R ≠ 0 then goto cat err;
     arn d14   , tl d3.7-16;   newfree:=
     tk-d3.7+16, ar  1d14  ;     bottom free + if free on disc then
     tl  -16   , tln 16    ;       bits (24, 27, first free) else 0;
     gr  b-2   , grn b-4   ;   sizes:= 0; LA:= t;

a12: qq (c15)  t  -1   LA  ; get entry: if LA then i word:= iword - 1;

a13: pp  0     , hs  a2    ; start entry: p:= 0; get next word;
     ga  b12   V       NZ  ;   if R ≠ 0 then save bits:= Raddr
     hv  a13           NB  ;   else if not end of catalog then goto start entry
b7:  hv [finish]       LZ  ;   else goto finis;
     hs  a2                ;   get next word;
a14: gp  b13   , hs  a2    ; loop entry: entry sz := p; get next word;
     grn pb-1  V       LA  ;   if it is an area word then
     hv  a14           NZ  ;     word [entry buf + p - 1]:= 0
     grn pb-1          M   ;   else if R = 0 then goto loop entry;
b8:  pi[mainin], ppn b     ;   word[entry buf + p - 1] M := 0;
     arn 1.7   D       LQB ;   indicator:= main in; p:= entry buf;
     ab  b     , gr  b     ;   if force sum then set sumbit in area word;
     pm  b-2   , hs  a     ;   M:= new free; sum and move;
     pp  b                 ;   p: entry buf; sum ok:= R = 0
b9:  hsr[main action]  IZB ;   main action;
```

```
a16: pp (b5)   Vt  1   LTA ; next entry: if -, namelist then
     hv  a12           M   ;   begin LA:= t; goto get entry end;
     arn p     , tl  -6    ;   for param addr:= paramaddr + 1 while
     ca  -1    , hv  a16   ;     not last name word do;
                           ;
a17: hsn a7                ; get name entry: test param;
     hv (b7)   , hs  c1    ;   if last param then goto finis else search;
     hv  a3            NZ  ;   if R ≠ 0 then goto cat err;
     gm  b-4           MPC ;   save spec MPC:= M;
     hs  a19           LZ  ;   while R = 0 do back up;
     hs  a19           NA  ;   while NA do back up;
     ps  a12   , hv  a19   ;   set return(get entry); goto back up;


a18: pa  b11   t   a21     ; set write back up: changed:= t;
a19: arn c15   , ca (c7)   ; back up: if i word = place 0 then
a20h:                      ;     goto if changed then write get previous
b11: hv a22[changed],gac15 ;       else get previous track;
     pmn(c15)  Xt  -1      ; backwards: i word:= i word - 1; M:= 0;
     hr  s                 ;   R:= store[iword]; return same;

a21: hs  a23               ; write get previous: sum and write;
a22: arn(c4)   Dt  -1  M   ; get previous track:
     ca  -1    , ac  c5    ;   track:= track - 1;
     pa  c4    t  959  NB  ;   if track = -1 then
     qq (c8)   t  -2       ;     begin group:= group - 1; track:= 959 end;
     hs  c4                ;   rel track:= rel track - 2; select and read;
     arn c9    , hh  a20   ;   i word:= i sum; goto back wards;

a23: hsn c8                ; sum and write: sum track;
     pan b11   Xt  a22     ;   changed:= f; M:= R:= 0;
     sk (c7)   , hrn s1    ;   to drum (place 0); return;

a24:                       ; main set sum:
b12: pm[save bits]XD   IOB ;   if -, sumit then
     pi  1.1   VXt 767 LRB ;     begin sum ok:= f; restore bits of
     ga  b     , hv  a27   ;     area word; goto call print end;
     sc  1b    , hs  a27   ;   sum ok:= t; sum word:= sum word - R;
     hs  a19           NA  ;   call print; while NA do back up;
     arn 1b    , gr (c15)  ;   store new sum word;
     ps  r1    , hv  a18   ;   write back up;
     arn b     , gr (c15)  ;   store new area word;
     ps  a16-1 , hv  a23   ;   set return next entry; goto sum and write;

a26: pi  1.1   t   767 NRB ; main list: if -, sum it then sumok:= t;
a27: gi  b17   , hv  a55   ; call print: result in:= in; goto print entry;
```

```
a32: pp  0     , hh  a34   ; fin compr: p:= 0; goto move entry;

a33: arn b     , ck  -16   ; main compr:
     tk  16    , ar  b-2   ;   if reserved bit then
     gr  b             LPB ;     area word:= (bits 0, 23, area word) pos 23
     gm  b-2   , hs  a26   ;      + newfree; newfree:= M; main list;
     arn(b13)  D           ;   sizes:= sizes + entry sz pos 23;
     ck  -14   , ac  b-4   ;   p:= entry size;
a34h:                      ;
b13: pp[entrysz], it b-1   ; move entry: i entry:= entry buf - 1;
a35h:pa  b14   , pp  p-1   ; loop move: p:= p - 1;
b14: arn[i entry] t 1  IPC ;   i entry:= i entry + 1; i out:= i out + 1;
b15: gr 39b[i out] t 1 MPC ;   word [i out] with marks:= word [i entry];
     ar  2     D       LA  ;
     ar  1     D       LB  ;   word [out buf + 39]:= word [out buf + 39] -
     sc  79b   , arn b15   ;     word [i entry] - marks pos 9; LB:= f
     nc  78b   , hv  a36   ;   if i out = out buf + 38 then
     pm  b-1   , hs  a1    ;     begin
     sk  40b   , arn a37   ;     to drum (outtrack, outbuf);
     ac  b-1   , ud  a10   ;     out track:= out track + 1;
     vk (b2)   , ud  1a10  ;     i out:= out buf + 39; rel cat := relcat + 1;
     pa  b15   t   39b     ;     word[i out] MC:= -rel cat - 3;
                           ;     LB:= t; end;
a36: bs  p     , hh  a35   ;   if 0 < p then goto loop move;
     bs  p1    , hv  a16   ;   if p = 0 then next entry;
     hv (b14)  Dt  -1  NB  ;   if -, LB then
                           ;     begin i entry:= i entry - 1; goto loop move end;
     hs  c2                ; modify free: get free;
     arn d14   , sr  b-2   ;   R:= bits(24, 39, free area word) -
     tl  -16   , tln 16    ;     new free;
     sc  d14   , ck  16    ;   free area word:= free area word - R +
     ac  d14   , hs  a23   ;     R pos 23; sum and write;
     pp  a38   , hs  a58   ;   print text (ǂ<words in cat, free sizeǂ);
     pm  d14   , tl  7     ;   M:= sizes + bits(8, 23, free area word);
     tln 16    , ar  b-4   ;   R:= area print format[0];
     pm  a56   X       IRA ;
     gm  b-4   , hs  a57   ;   print group;
     sy  64    , hv  a42   ;   writecr; goto exit;

a37: m 1
a38: t
Words in cat, free size   ;
```

```
a40: arn s     , tk  10    ; alarm: mess:= part 2 [word[s]];
     ga  b18   , tk  20    ;   R:= part 4[word [s]] pos 9;
b17: a41=i-1               ;
a42: pi[result in], vk d19 ; exit: in:= result in;
     pm2 c22   X       IZA ;   callok:= LZA:= R = 0;
     tk  10    , vk  25d16 ;   if part 2(current output) = 1023 then
     nc  -1    , hv  a45   ;     begin
     lk  d14   , vk  25d16 ;     to core (last image track, work buf);
     hv  a44   X       NZA ;     if call ok then
     pmn b-4               ;       begin
     hv  a44   X       NTA ;       if namelist then
     qq        X       NC  ;       image M:=
     qqn               LT  ;       if spec present then save spec else 0;
     gr30 d14  , arn b     ;       R:= area word; end;
a44: gr  32d14 , tl  -39   ;     image R:= R;
     gr  34d14 , gi  27d14 ;     image in:= in
     arn d13   , nc  0     ;     if called from program then
     sk  d14   , vk  26d16 ;     to drum (last image track, work buf);
     pt  b18   t   1c24    ;     set alarm print to normal return;
                           ;     end
a45: acn -1            M   ;   full inhibition;
     vy  0     t   -1d18   ;   no by inhibition;
b18:                       ;
b18h:ncn[mess] , hs  c24   ;   if mess ≠ 0 then alarm print (mess);
     hv  -9                ;   goto call help;


[actiontable]              ; act: main action, start, finis
a50=i-a61                  ; no name list:
qq a26-b9 t a11 + a42.19   ;  0: main list, start no name, exit;
qq        t a8             ;  1: -, param err, - ;
qq a33-b9 t a10 + a32.19   ;  2: main compr, start compr, fincomptr;

a51=i-a61                  ; name list:
qq a26-b9 t a17 + a42.19   ;  0: main list, get name entry, exit;
qq a24-b9 t a17 + a42.19   ;  1: main set sum, get name entry, set sum;
qq        t a8             ;  2: -, param err, - ;

a55: [first word of print entry]
```

```
[print entry

   Outputs a catalog entry on the media given in by.
 The format of the output is determined by the entry, a preset
 printkind, and, a boolean sumok
   printkind may be 0, 1, 2, or 3 with the following
 effect:
   0. if sum ok then no output else as printkind 1;
   1. Only kind and names will be output.
   2. The entry will be output in the form of a complete
      call of set or, if the area is reserved, res.
   3. The whole entry will be output in a form corresponding
      to the parameter format to set, but with two additional
      possible lined letters:
       x meaning special bit present. If so then the secondary
       word, if present, will be printed as
       <bit 0 to 7> , <bit 8 to 23> , <bit 24 to 39>
       r meaning reserved bit present.
   Any output will be followed by:
      if -, sum ok then message(¢<sum¢)

 Call of print an entry:
   p:= address of first word of the entry. The entry must
     be terminated with a zero;
   print kind in address part of first word of print an entry
   LZB:= sumok.
   Indicator bits TB - QB must hold bits 3 -7 of the area word.

   hs first word

 ZA -QB are untouched upon exit. M, p, RA and RB have been used.
 R = 0 with no marks.
 Returns to s1.]


b a45, b15                 ; core block print entry;

a1:  is[print kind],pmsa44 ; printentry: R:= action table[print kind];
     ptn b3    VXt a18 LA  ;   area pr:= part 1(R);
     grn a12   X           ;   bit pr:= part 2 (R);
     ga  b1    , gt  b2    ;   spec pr:= part 3(R);
     ck  20    , ga  b4    ;   CRact:= part 4(R); save s:= s
     gt  b14   , gs  b5    ;   if printkind = 2 then resbit:= skip single
     arn p     , cl  -7    ;     else set no < print;
     ga  b15   , ps (b15)  ;     comment no r should be output in
     pm  p     , ca  0     ;     printkind 2;
     pa  b15   t   16      ;   M:= area; kind:= s:= bits(0, 2, M);
     arn sa            IRC ;   if kind = 0 then kind:= digit zero;
b1:  hv [area pr]          ;   R:= formats[s]; point sep:= LRA:= LA
                           ;   terminated:= LRB:= f;
                           ;   goto area pr;
```

```
[area print format[kind]                    ;
a:[0]qq  8.4+31.9+31.14                     ;
[1]  qq  8.4+31.9+19.14+27.19               ;
[2]  qq  8.4+31.9+ 4.14+17.19+21.24+19.29   ;
[3]  qq  8.4+31.9+19.14+20.19+22.24         ;
[4]  qq  8.4+17.9+25.14+25.19+25.24,        ;
[5]  qq  8.4+17.9+25.14+25.19+25.24,        ;
[6]  qq  8.4+17.9+25.14+25.19+25.24,        ;
[7]  qq  8.4+17.9+25.14+25.19+25.24,        ;

[8]  qq  8.4+31.9              ; print format: reserved area, printkind = 2.
[9]  qq 23.4+31.9+31.14        ;   -      -  : secondary word, special.
[10] qq 25.4+25.9+25.14+25.19, ;   -      -   sumword or spec.

[11] m  10-8   ;
[12] m     10 ;


a2:  t   sum;  ;

b:   qq [number];
[1]  qq [format];

a4:  arn 8a            LPB ; area pr 2: if reserved then R:= reserved format;
a5:  ps  a8    , hv  a20   ; area pr 3: set return(start bit pr); goto CRgroup;
a6:  hh  a13           LZB ; area pr 0: if sum ok then goto exit;
a7:  a8=a7-1               ;
b2h: arn p     , hv [bitpr]; start bit pr: R:= area; goto bit pr;

a9:  qq 23 [x] , hs  a15   ; bit pr 23: start single (x);
     qq 39 [p] , hh  a16   ;   next single (p);
b3h: qq 41 [r] , hha16[resbit];   resbit (r);
     qq 57 [i] , hh  a16   ;   next single (i);
     qq 18 [s] , hh  a16   ;   next single (s);
     pm  p1    , ck  -1    ;   M:= secondary word;
     hv  a11           NA  ;   if is secondary then
     qq        V       LO  ;     begin if -, sumbit then
     qq 52 [d] , hs  a17   ;       print single (d);
     arn 9a    V       LTB ;     if special bit then R:= sec special format else
a10: arn 10a               ; spec pr 23: R:= four bytes format;
     hs  a21           IRA ;     point sep:= LA; comma group;
                           ;     end;
a11: pp  p1    , arn p     ; name pr: p:= p + 1; R:= word[p];
     hv  a11           LA  ;   if LA then goto name pr
     hv  a12           LZ  ;   if R = 0 then goto end print;
b4:  hv [specpr] X     NT  ;   if spec then goto spec pr;
     hs  a35               ;   newline;
     ps  a11-1 , hv  a31   ;   set return (name pr); goto printtext;
```

```
a12: sy  64    , sy  17    ; finis: if printkind = 2 then
     pp  a2                ;   print end help call;
     hs  a31           NZB ;   if -, sum ok then print text(‹<sum›);
a13h:sy  64    , arn r-1   ;   writecr;
b5:  ps[saves] , hrn s1    ; exit: s:= saves; R:= no marks 0; return
a15: ck  3                 ; start single: R:= R shift 3;
a16h:hh  a18           NO  ; next single: if bit(0, R) = 1 then
a17: hs  a39           NRB ; print single: begin if -, terminated then comma;
     sy  14[_] , sy (s)    ;   print lined letter from cell s followed by space end;
a18h:sy  0     , ck  1     ; skip single: R:= R shift 1;
     ps  s1    , hv  s     ;   s:= s + 1; go back;

a20: hs  a35               ; CR group: new line;
a21: hs  a39           NRB ; comma group: if -, terminated then comma;
a22: hs  a25           IRB ; print group: terminated:= f; print next;
     hr  s1            LZ  ;   if R = 0 then return;
     sy  59[.] V       LRA ;   if point sep then print point
     sy  27[,] , sy  0     ;   else print comma and space;
     hv  a22               ;   goto print group;

a24: hs  a28               ; skip it: skip;
a25: gm  b     , tk  -5    ; print next: number:= M;
     ga  b7    , tk  10    ;   no of bits:= bits(0, 4, R);
     gr  1b    , xrn       ;   format:= bits(5, 39, R) pos 34; R:= M;
     bs (b7)   , hv  a24   ;   M:= 0; if no of bits < 16 then goto skip it;
b7:  cl [no of bits] t 17  ;   no of bits:= no of bits - 15;
                           ;   M:= bits(0, no of bits, R);
a26: gs  b10   , mln 11a   ; print M:
a27: mln 12a   , it  128   ;   comment prints M with layout ‹d›
b8:  ca 0[count], pa  b10  ;   maximum 8 digits, no sign;
     arn 16    DV      LZ  ;
     ck  -10   , pa  b10   ;
b9:  ga [digit]D           ;
b10: can[yes]  , sy (b9)   ;
     ncn(b8)   , hv  a27   ;

a28: arn b     , pm  1b    ; skip: M:= bits(no of bits, 39, number);
     tk (b7)   X           ;   R:= format;
     hr  s1                ;   return;


a31:
a30h:pmn p     , cln -6    ; print text:
     ck  -4                ;
     pi  0.9   Xt  1022    ;   M:= word[p];
     hh  a32   X       NZ  ;   print string in M
     xr        , ca  7     ;
     pp  p1    , hv  a31   ;   if continued then
a32h:hr  s1    , ga  b12   ;     begin p:= p + 1; goto print text end;
     ca  10    , hr  s1    ;   terminated:= f;
     ca  63    , it  1     ;   return;
b12: sy [char] , hhn a30   ;
```

```
a35: sy 64 [CR], bs (b15)  ; newline: print CR; if kind not printed then
b14h:sy  58    , hv [CRact];   begin print LC; goto CR act end;
     sy  0     , sy  0     ;   four spaces; terminated:= t; return;
a36h:hh  a40   , sy  18 [s]; set: print text(¢<set¢);
     sy  53 [e], sy  19 [t];   print comma;
     hs  a39               ;
a37:a38h:                  ;
b15: sy [kind] , pa  b15   ; kind pr: outchar(kind); kind:= 0;
a39:                       ;
a40h:sy  27[,] , sy  0     ; print comma: print comma and two spaces;
     pi  1.9   t   1022    ;   terminated:= t;
     sy  0     , hr  s1    ;   return;
a41: hh  a36           NPB ; set or res: if -, reserved then goto print set;
     sy  41 [r], sy  53 [e];   printtext(¢<res¢);
     sy  18 [s], hh  a38   ;   goto print comma;

[Action table[print kind]          ; area pr  bit pr   spec pr  CRact
a44: qq  a6.9+a11.19+a11.29+a37.39 ; area pr0,namepr   namepr   kind pr
[1]  qq a11.9+a11.19+a11.29+a37.39 ; namepr  ,namepr   namepr   kind pr
[2]  qq  a4.9+ a9.19+a10.29+a41.39,; areapr2 ,bitpr23  specpr23 set or res
[3]  qq  a5.9+ a9.19+a10.29+a37.39 ; areapr3 ,bitpr23  specpr23 kind pr


[Define names for use in outer block]
a56=a, a57=a22, a58=a31    ; printformat 0, print group, print text.

e [print an entry]         ;
```

```
i=39i, d=k-d1              ;
b k=d42, i=0, a10          ;
a1=d19-960                 ; a1 = group no for image.
a2=a59, a3=a60, a7=a61     ; define entry points.
a8=a62, a9=a63             ;
a=d                        ; a = no of blocks.
<d35, a=1>                 ;

<d39                       ; if aux only then
i=d2                       ;
     hs1                   ; begin
     hv  a4                ;   (if aux reserved then
<d36                       ;
     tres;                 ;   res, no of blocks
     qqf a.39              ;   else
×                          ;
<d39                       ;
     tset;                 ;    set, aux kind, no of blocks,
     qqf d35.39            ;    typein)
     qqf a.39              ;
[after i follow STOP, SUM, a, and a SPACE]
ia base check, compress, list, setsum
s
[STOP, CLEAR]
>                          ;
<d39                       ;
     qq  39,               ; concat pid 0,
     qq  57,               ;
     qq  52,               ;
     qqf                   ;
     tcheck;               ; check,
     qqf d.9+a3.19+a2.29   ; spec,
     tlist;                ; list,
     qqf d.9+a7.19+a2.29   ; spec,
     tsetsum;              ; setsum,
     qqf d.9+a8.19+a2.29   ; spec,
     tcompress;            ; compress,
     qqf d.9+a9.19+a2.29   ; spec<
     qqf,                  ;
a4:  hs  1                 ;
     hv  a5                ;
     tmove;                ; move, b loadplace, check<
     qq  50,               ;
     qqf d.23+d1.39+a1.29-a1.33 ;
     tcheck;                  ;
     qqf,                  ;
a5:  hs  1                 ;
     hv  a6                ;
     tsetsum;              ;
     tcheck;               ;
     qqf,                  ;
d2=i                       ;
a6:  hsf 2                 ; end
```

```
×                          ; else
i=d48                      ; load to primitive catalog
     qq d35.2+11.7+d36.5+d.23+d1.39, ;
     qq,                           ;
     tcheck;               ;
     qq d.9+a3.19+a2.29    ;
     tlist;                ;
     qq d.9+a7.19+a2.29    ;
     tsetsum;              ;
     qq d.9+a8.19+a2.29    ;
     tcompress;            ;
     qq d.9+a9.19+a2.29    ;
d48=i                      ;
     qqf                   ;
>                          ;

d1=d+d1                    ;
e [image load]             ;
e [check ff]               ;
[After i follow STOP, SUM, a, and a SPACE]
ia check, compress, list, setsum
s
```

[STOP, CLEAR]

```
b k=d1, i=40d13, a67, b17  ;
[variables]
b:   qq                    ; length
[1]  qq d3.2+1.5,          ; area, initially reserved in free, for use by res.
[2]  qq         ,          ; sum word
[3]  qq                    ; start point test value

[kind table]

[-1] pi   4    , hv  a4    ; kind 4 to 7
a:   pi   2    , hv  a12   ; kind 0
[1]  pi   0    , hv  a40   ; kind 1
[2]  pi   0    , hv  a41   ; kind 2
[3]  pi   0    , hv  a42   ; kind 3

[constants]
[4]  qq         2d38.39    ; top core + 1 for programs
[5]  qq           40.39    ; track length
[6]  qq -1.7+1.23-1.39     ; mask for zeroing booked
[7]  qq      7.27          ; unit for caroussel
```

```
a10: pp  d13   , arn p1    ; set:
     ck  -10   , ga  b4    ;   p:= addr of first param - 1;
     ck  7     , gr  1b    ;   R:= param[p+1];
     pa  b4    t   -1  LO  ;   kind:= if R < 4 then R else -1;
     hsn a2                ;   area:= R pos 2; R:= 0;
     qqf             7.39  ;   length:= test add next(0, 7, true);
     gr  b     , hsn a2    ;   test add next(dum, dum, false);
b4:  hv [kind] t   a       ;   go to table[kind];

a2:  hv  a6            LA  ; procedure test add next(lower, upper, test);
     hv  a6            NB  ;   comment test = word[s1] is f marked,
     ac  1b    , pp  p1    ;    lower = -signed bits(0, 19, word[s1]),
     arn s1    , tl  -20   ;    upper = bits(20, 39, word[s1]);
     ar  p     V       LB  ;   begin if paramkind(R) ≠ help number
     arn p1    , hv  s1    ;     then go to param err;
     hv  a5            LT  ;   area:= area + R; p:= p + 1;
     tln 20    , sr  p     ;   if -,test then begin R:= param[p+1]; return end;
     hv  a5            LT  ;   if param[p] < lower  ∨ param[p] > upper
     arn p1    , hv  s2    ;     then go to value err;
                           ;   R:= param[p+1]; return;
a11: arn r1    , hs  c2    ; res: R:= not zero; get free;
     pm  1d14  , tln 7     ;   p:= addr of first param -1
     tln 16    , pp  d13   ;   if param kind[p+1] ≠ help number then
     pm  p1                ;     length:= booked
     gr  b     V       NB  ;   else begin length:= R:= param[p+1];
     gm  b     , hsn a2    ;     R:= 0; test add next(dum, dum, false);
     pi1.3+1.8-d3.8, srn b ;   reserved:= t; kind is 0:= free kind = 0;
     hv  a7            NT  ;   if length ≤ 0  ∨ length > free size
     pm  d14   , tln 7     ;     then go to length err;
     tln 16    , sr  b     ;   area:= area + first free +
     hv  a7            LT  ;     length pos 23;
     arn b     , tl  16    ;   go to get bits and sum;
     ac  1b    , hv  a14   ;

a4:  arn b                 ; kind 4 to 7:
     ac  1b    , tl  -2    ;   area:= area + length;
     ca  0     , hv  a14   ;   if bits(0, 7, length) = 0 then
                           ;     go to get bits and sum;
a5:  hs  a65   , qq  sa8   ; value err: alarm(¢<value>, 1);
a6:  qq  e5    , hs  c24   ; param err: alarm print(¢<param>);
a7:  hs  a65   , qq (a9)   ; length err: alarm(¢<length>, 4);

a8:  tvalue;               ; texts.
a9:  tlength;              ;
```

```
a12: gr  3b    , hs  a2    ; kind 0: start point test value:= R;
     qqf     d22.31+d22.33 ;   test add next(0, max start kind 0 + 1, true);

a13: arn b     , ck  16    ; test size:
     ac  1b    , srn b     ;   area:= area + length pos 23;
     hv  a7            NT  ;   if length + start point test value ≥
     sr  3b    , ar  s1    ;     last used value of upper then
     hv  a7            LT  ;     go to length err;

a14: arn p1    , ps  2     ; get bits and sum: R:= param[p+1];
     hv  a6            LC  ;   if param kind(R) = lined then
     hh  a16           NA  ;     begin p:= p + 1;
     pp  p1    , ca  18 [s];     if R = s then
     pt  b11   , hv  a15   ;       begin sum bit:= 1; go to get sum end;
     ca  39 [p], pan b10   ;     if R = p then programbit:= 1
     ca  57 [i], ptn b10   ;     else if R = i then inhibit bit:= 1
     hv  a14           LZ  ;     else if R = d then get sum:
     nc  52 [d], hv  a6    ;       begin a words:= 2;
a15: pm  p1    , gm  2b    ;         sum word:= param[p+1]; R:= 0;
     gs  b14   , hsn a2    ;         test add next(dum, dum, false) end
                           ;     else go to param err; go to get bits and sum
a16h:hv  a14   , gp  b15   ;     end;
b10: ns  1.4   , is  s-1.6 ;   j:= p; R:= programbit pos 4 +
b11: arns1.4+1.6+1.7Dt-1.7 ;     inhibit bit pos 6 + sum bit pos 7;
     ac  1b    , nc  0     ;   area:= area + R; if R ≠ 0 ∧
     hv  a6            LQB ;     kind is 4 to 7 then go to param err;

a17: hs  c1                ; test name: search;
     hv  a36           NT  ;   if R ≥ 0 then go to cat err res set;
a18: arn p1    , tl  -6    ; skip name: R:= param[p+1];
     ca  -1    V       NC  ;   if param kind(R) ≠ name then go to param err;
     hv  a6                ;   if bits(0, 3, R) = 15 then
     pp  p1    , hv  a18   ;     begin p:= p + 1; go to skip name end;
     nc  -6    , hv  a5    ;   if bits(0, 3, R) ≠ 10 then go to value err;
     acn p1            MB  ;   set f mark on last name word;
     pp  p1    , pmn p1    ;   p:= p + 1; M:= 0; R:= param[p+1];
     hh  a22   X       NC  ;   if param kind(R) = name  ∨
     hh  a22   X       LC  ;     param kind(r) = finis then
     hv  a6    X       LA  ;       go to test area program;
                           ;   if param kind(R) = lined then go to param err;
```

```
     gm  p1    X       M   ; test spec:
     hv  a5            LZ  ;   clear marks on spec word;
     hv  a5            LT  ;   if r ≤ 0 then go to value err;
     cl  10    , ck  20    ;   M:= tracks part(R); p:= p + 1;
     tk  30    , pp  p1    ;   Raddr:= first core part(R); go to test progr;

a22h:hv  a23   , pm  b     ; test area program:
     arn 40d13 D           ;   if kind is 0 then
     hh  a24           NRA ;     begin M:= length; Raddr:= fixed first core;
a23: ck  10    , sr  4a    ; test program: R:= Raddr - top program core;
a24: ml  5a    , pm  p1    ;   RM:= R + M × 40;
     ca (b10)  , hv  a5    ;   if program ∧ RM ≥ 0 then go to value err;
                           ;     end;
     hv  a17           NA  ;   if param kind[p+1] = name then go to test name;
     hv  a6            NB  ;   if param kind[p+1] ≠ finis  ∨
     bs p-36d13, hv  a6    ;     too many params then go to param err;

a25:                       ; store: comment LRC = fill up;
b14: bt  1 [awords] t -1   ;   awords:= awords - 1; if awords ≥ 0 then
     arn b     Vt  1   IRC ;     begin b:= b + 1; R:= set RC(word[b]) end
b15: arn[j]    t   1   IRC ;   else begin j:= j + 1; R:= set RC(word[j]) end;
     qq (b15)  Vt  -1  LRC ;   if fill up then
     gr (c15)  V       MRC ;     begin j:= j + 1; store[iword] MB:= 0 end
     grn(c15)          MB  ;   else store[iword] MRC:= R;
     arn(c15)  Dt  1   IZA ;   iword:= iword + 1; found:= t;
     nc (c9)   , hv  a25   ;   if iword ≠ i sum then go to store;
a26: pa  c6    , hs  c8    ; full track: mode:= 0; sum track;
     hv  a27           LRC ;   if -, fill up then
     arn 1c    , tl  -2    ;     begin
     ca (c8)   , hv  a34   ;     if rel track = max cat tracks then
     sk (c7)   , ps  a25-1 ;       full catalog;
     qq (c8)   t   1       ;     to drum(place 0); rel track:= rel track + 1;
     hv (c4)   Dt  1       ;     track:= track + 1; select track; go to store
                           ;     end;
a27: sk (c7)   , vk (c4)   ;   to drum(place 0); wait drum;

a28: hhn a66           NTB ; set new free:
     arn r1    , hs  c2    ;   if -, reserved then go to exit;
     arn b     , ac (c7)   ;   Raddr:= not zero;
     tk  16    , sc (c7)   ;   free word:= free word + length - length pos 23;
     arn 1d14          ITB ;   reserved:= f;
     mb  6a    , gr  1d14  ;   booked:= 0;
     hv  a26               ;   go to full track;

a30: tname;                ;
a31: tno clear;            ;
a32: ca  -1                ; cat err clear: if Raddr = -1 then
     hs  a65   , qq  sa30  ;   alarm(¢<name>, 1) else
a33: qq  e7    , hs  c24   ;   alarm print(¢<catalog>);
a34: hs  a65   , qq  pe2   ; full catalog: alarm(¢<full>, 3);
a35: hs  a65 ,  qq a31+2.29; clear err: alarm(¢<no clear>, 2);
a36: nc  0     , hv  a33   ; cat err res set: if Raddr = 0 then
     hs  a65 ,  qq a30+2.29;   alarm(¢<name>, 2) else alarm print(¢<catalog>;
```

```
<d41 [if buffer media then begin]

a40: ck  12    , hs  a2    ; kind 1: R:= disc unit pos 27;
     qqf -8.19      +15.39 ;   test add next(8, 15, true);
     gr  3b    , hs  a2    ;   start point test value:= R:= first disc block;
     qqf      d52.29+d4.39 ;   test add next(0, max disc blocks, true);
     hv  a13               ;   go to test size;

a41: ck  10    , hs  a2    ; kind 2: R:= grouped pos 29;
     qqf -1.19      + 3.39 ;   test add next(1, 3, true);
     ck  4     , gr  3b    ;   start point test value:= R:= reel pos 35;
     hs  a2                ;   test add next(0, 63, true);
     qqf             63.39 ;   start point test value:=
     ac  3b    , arn 7a    ;     start point test value + caroussel block;
     ar  p1    , hs  a2    ;   R:= 7 pos 27 + caroussel block;
     qqf             15.39 ;   test add next(0, 15, true);
     hs  a13               ;   set last used upper to 1024;
     qq            1024.39 ;   go to test size;

a42: ck  12    , hs  a2    ; kind 3: R:= tape unit pos 27;
     qqf -1.19      + 6.39 ;   test add next(1, 6, true);
     ck  7     , hs  a2    ;   R:= file pos 32;
     qqf             31.39 ;   test add next(0, 31, true);
     gr  3b    , hs  a2    ;   start poin test value:= R:= first tape block;
     qqf            127.39 ;   test add next(0, 127, true);
     hs  a13               ;   set last used upper to 8000;
     qq            8000.39 ;   go to test size;

×    [end else]            ;

a40=c57, a41=c57, a42=c57  ; kind 1: kind 2: kind 3: alarm print(¢<kind¾);

>                          ;
```

```
a50: arn 6a    , gr  2c    ; clear: area word:= something with special bit;
     pp  d13   , hs  c1    ;   p:= address of first param - 1; search;
     hv  a32           NZ  ;   if R ≠ 0 then go to catalog err;
     arn 2c    , ck  3     ;   if special bit is set in area word then
     hv  a35           LO  ;     clear err;
     pi  327   t  48       ;   finis:= LRC:= reserved:= t; R:= 0;
     tk  3             ITB ;   changed:=f; if -, reserved bit then
     hvn a51           LTB ;     begin reserved:= f; if kind = 0 then
     tk  35                ;
     hvn a51           NZ  ;       for R:=R - store[iname:=iname + 1]
[-1] arn 2c    , ud  c19   ;         while NC do
     hv  r-1           NC  ;     if R ≠ 0 then clear err;
     hv  a35           NZ  ;     end;
                           ; search end entry:
a51: pm (c15)  , ps  a51-1 ;   M:= store[iword]; set return(search end entry);
     hv  a32           NZ  ;   if R ≠ 0 then go to cat err clear;
     qq        X       NA  ;   if -, areaword then swop;
     hv  c15           NZ  ;   if R ≠ 0 then go to get word;
     qq                IQB ;   LQB:= LB;

     ps  r1    , hv  a57   ; clear entry: back up;
     hs  a56           NPA ;   while NPA do clear back up;
     hs  a56           LPA ;   while LPA do clear back up;
     hs  a56           LZ  ;   while R = 0 do clear back up;
     arn 2c    , ps  a28-1 ; test if free can be changed:
     hv  a52           NTB ;   set return(exit);
     ck  -16   , ar  2c    ;   if reserved ∧ first track(free) =
     sr  c     , cl  24    ;     first track(area) + size(area) then

     hv  a52           NZ  ; get new free descr:
a60: hs  a57           NPA ;     begin skip to area: while NPA do back up;
     hs  a55           LPA ;     while LPA do set area;
     arn 2c    , tk  4     ;     if bit(3, area word) = 1 then
     tk  2     V       NT  ;       R:= free word 1
     arn 1c    , hv  a61   ;     else if bit(5, area word) = 0 then
     hv  a60           NT  ;       go to skip to area
     ck  -22   , ar  2c    ;     else R:= area word + bits(8, 23, area word;
a61: sc  c     , arn c     ;     length:= bits
     tl d3.7-16,tln-d3.7+16;       (24 + free kind × 4, 39, R) - first free;
     sc  b     , ps  a28-1 ;     set return(set new free)
                           ;     end;

a52: hr  s1            LZB ; maybe write: if -, changed then return;
a53: hsn c8            IZB ; sum and write: changed:= f;
     sk (c7)   , hr  s1    ;   sum track; to drum(place 0); return;
```

```
a54: hs  a53          NZB ; get previous track: if changed then sum and write;
     arn(c4)   Dt  -1  M   ;   track:= track - 1;
     ca  -1    , ac  c5    ;   if track = -1 then
     pa  c4    t  959  NB  ;    begin group:= group - 1; track:= 959 end;
     qq (c8)   t   -2      ;   rel track:= rel track - 1;
     hs  c4                ;   select track and read it;
     arn c9    , hh  a58   ;   i word:= isuum; go to backwards;

a55: gr  2c    , hv  a57   ; set area back up: area word:= R; go to backup;

a56: grn(c15)         MQB ; clear back up: store[iword] MQB:= R:= 0;
     pi  0.1   t  767     ;   changed:= t;

a57: arn c15   , ca (c7)   ; back up: if iword = place 0 then
a58h:hv  a54   , ga  c15   ;     go to get previous track;
     pmn(c15)  Xt  -1  IPC ; backwards: iword:= iword -1; R:= store[iword];
     hr  s  [ not s1 ]     ;   return same;

a65: arn s     , tk  10    ; alarm: set text addr
     ga  b17   , tk  20    ;   Raddr:= exit value;

a66: pm  2c22  X           ; exit:
     ck  10    , vk  d19   ;   if mask current output = -1 then
     nc  -1    , hv  a67   ;     begin
     vk  25d16 , lk  d14   ;     store exit value in R cells
     vk  25d16 , gm 32d14  ;     in image;
     grn 35d14 , arn d13   ;     if called from programthen image to drum;
     nc  0     , sk  d14   ;     set alarm print to normal return
     pt  b17   t 1c27      ;     end;
b17: b17h:                 ;
a67: ncn 0     , hs  c24   ;   if text addr ≠ 0 then alarm print(text addr);
     hv  -9                ;   go to CALL HELP;
```

```
i=39i, d=k-d1       ; d = no of tracks
b k=d42, i=0, a3    ;
a1=d19-960          ; a1 = image group
a=d, <d35, a=1>     ; a = no of blocks

<d39                ; if aux only then
i=d2                ;  begin
     hs  1          ;
     hv  a2         ;
     tmove;         ;  move, b load place, set <
     qq  50         ;
     qqf d.23+d1.39+a1.29-a1.33         ;
     tset;          ;
     qqf            ;

a2:  hs  1          ;
     hv  a3         ;
     tsetsum;       ;  setsum, set <
     tset;          ;
     qqf            ;
d2=i                ;
a3:  hsf 2          ;  end
×                   ; else
i=d48               ;  load to primitive catalog;
     qq  d35.2+11.7+d36.5+d.23+d1.39,   ;
     qq                                 ;
     tset;                              ;
     qq  d.9+a10.19+40d13.29            ;
     tres;                              ;
     qq  d.9+a11.19+40d13.29            ;
     tclear;                            ;
     qq  d.9+a50.19+40d13.29            ;
d48=i               ;
     qqf            ;
>                   ;

d1=d+d1             ;
e [load image]      ;
e [set,res,clear.]  ;

[after i follows STOP, SUM and a sum character]
ia set,res,clear
s
```

Order No.: 490

Annotated Help 3 Programs

vol. II

edit	20 pages
exit	2 –
move	8 –
print, pair	20 –
run	2 –
slip	25 –
start	2 –
system punch	11 –
check bin	2 –
cattap	1 –
punch head bin 0	1 –
punch head kompud	4 –
create new -> old	1 –

December 1967

```
[18.7.67]   [edit/reservation of buffer areas]   [page 1]
            [character table]
[STOP, CLEAR]
b k=d1, i=90, b60, e60     ; edit, outermost block

<d41, e1=-166,× e1=10      ; input buffer in Pass 1-2
>                          ; used by line buffer, reversetracks,
                           ; INIT MEDIUM in Pass 1
<d41, e2=-86 ,× e2=-86     ; byte input buffer in Pass 2
>                          ; INIT MEDIUM in Pass 1
<d41, e3=10  ,× e3=-166    ; output buffer in Pass 1-2
>
[ marks in e3-79e3 are initialized at start pass 1 as follows:
     39e3   A   mark
     79e3   C   marks, all others are zeroes]

[ indicator usage: in Pass 1              in Pass 2
    LZA       N  in2nd                     freely used
    LZB       L  endOK                        -
    LTA          not used                     -
    LTB              -                  N  case free
    LPA          A mark of tableword    N  only input sum
    LPB          B   -         -        N  copy
    LQA       N  underlined             X  external output
    LQB       N  sum                       freely used
    LRA           not used                  blind
    LRB       X  typewriter input        N  ready
 where L,N or  X indicates initialization. Used compound conditions:
                  NZC, LPC                  NZC, LPC, NRC                ]
e4:
b a5
[ character table:
     The i-th entry in the following table describes the type of the cha-
  racter with value i. The entries can be freely changed by appropiate
  parameters. (cf. table change, page 12) Changing the description for
  LC, UC, TF or <10> can, however, interfere with the internal logic.
                 description     parameter for edit
    normal       qqf             l <chv>, n
    alarm        qq      -2,     l <chv>, a
    blind        hvf  s1   ,     l <chv>, b
    skip         qq  512         l <chv>, s
    end          qq  770         l <chv>, e
    case indep.  qq  896         l <chv>, c
    replaced     qq  <chv>       l <chv>, r, <chv>
          where <chv> ::= <help number>
     The descriptions for LC, UC, <10>, CLEAR and SUM are initia-
  lized to execute special actions, not available for assignment to
  other characters]

a3:    pmf 0 , zqn(r+i+1)   ; constant for sum = true
[1]
```

```
e5:
[0]    hvf s1,
[1]    qqf
[2]    qqf
[3]    qqf
[4]    qqf
[5]    qqf
[6]    qqf
[7]    qqf
[8]    qqf
[9]    qqf
[10]   qq   832           ; TEN: goto trailing edge;
[11]   qq   770
[12]   hvf  s1,
[13]   qq   11
[14]   qqf
[15]   qq   -2,
[16]   qqf
[17]   qqf
[18]   qqf
[19]   qqf
[20]   qqf
[21]   qqf
[22]   qqf
[23]   qqf
[24]   qqf
[25]   qqf
[26]   qq   -2,
[27]   qqf
[28]   paf a48, hhf  a1   ; CLEAR: goto clearcode;
[29]   hvf  s1,
[30]   hvf  s1,
[31]   hvf  s1,
[32]   qqf
[33]   qqf
[34]   qqf
[35]   qqf
[36]   qqf
[37]   qqf
[38]   qqf
[39]   qqf
[40]   qqf
[41]   qqf
[42]   qq   -2,
[43]   qqf
[44]   hvf  s1,
[45]   qq   -2,
[46]   qq   -2,
[47]   qq   -2,
[48]   qqf
[49]   qqf
[50]   qqf
[51]   qqf
[52]   qqf
[53]   qqf
[54]   qqf
[55]   qqf
```

```
[56]   qqf
[57]   qqf
[58]   paf  b  , hvf s1    ; LC: case:= 0; blind action;
[59]   qqf
[60]   hvf  a  ,           ; UC: case:= 128; blind action;
[61]   pmf  s3 , hvf a2    ; SUM: goto sumcode;
[62]   hvf  s1 ,
[63]   qq   512
[64]   hvf  s1 ,

[ The following code up to 127e5 is treated as the continuation of the
  table therefore it must not contain any f marked word                ]

b:   qq [case]             ;
a2:  grn s3           MA   ; sumcode: save slow action:= slow action;
     gm  b26   , hv  s1    ;   slowaction:= check;
a:   pa  b     t   128     ;   return;
a1:  hv  s1    , pa  b1    ; clearcode: insum:= 0; sum:= true;
     hv  a4           LPC  ;   if first pass then blind action;
     pm  a3    , gm  b2    ;   insum2:= 0; A:= true;
e45: hv  a4           NQA  ; set summing: if -, external then begin
     pa  e8    Vt e25  NRB  ;   if -,ready  v caution then out:= outinternal and check
e47: pa  e8    t  e26  IPA  ;   else begin out:= outinternal; only input sum:= A;
a4:  hv  e35          M    ;   end end; blind action;

e
[ Start pass 2, stack updating:
     The string to be searched for is input from the drum and placed from
  e10 onwards. The string is terminated with the byte -512. Byte -511 in
  the input stream means the end of the corrections.                    ]

b a30

e6:  a: qq[count], pp 0    ; START PASS 2: nextstring: i:= 1;
a1:  pm (b15)  Xt  1       ; BYTE:
     hs  e33           NA  ;   stack[i]:= Raddr:= char from drum;
     gr  pe10          M   ;   marks[i]:= 0;
     pi (pe10) Vt -130 LT  ;   if Raddr ≥ 0 then begin i:= i + 1; goto BYTE end;
     pp  p1    , hv  a1    ;   ready:= Raddr = -511 ; case free:= false;
     pm  r             LPB ;   copy:= -, copy;
     bs  p             IPB ;   if i > 0 then begin
     acn pe9   V       MA  ;     set A mark(stack[i-1]); skip line end;
     hvn a11           NRB ;   if -, ready then goto STRING FOUND;
```

```
a2:  pp  1     V           ; BEGIN SEARCH: p:= 1; skip line;
a3:  hs (a4)           LRA ; PERHAPS BLIND: if blind then execute(action[char]);
                           ; blind action:

[ The next two instructions are changed if the input unit is the tape
  reader or the typewriter to:

e7:  lyn e18   V           ;   Raddr:= lyn; skip line;
[1e7|hv  e13   , xx        ; RESET AND START PRESSED: go to trailing edge;]

e7: [-3] pm e18 Xt 1       ; NEXT CHAR:
[1e7]hv  e22           LA  ;   Raddr:= input;
[-1] ga  a4    V       NT  ;   if parity  ∨ slowaction ≠ 0 then
     [changed to qq 0 if print or to qq 0, if check]
[3e7]hs  e36               ;   treat slow;
a4:b2: pm 0    Vt e5   IRA ; TEST: char:= Raddr + table base;
     [char] [sum]          ;   M:= set marks(cell[char]); blind:= Amark;
b1:  ac [insum]Dt  1       ;   if sum then insum:= insum + Raddr + 1;
a5:  hh  a13   X       NB  ; AFTER CENTRAL: if spec then go to TREAT SPECIAL;
e8:  hs  e26[out]      LPB ; OUTPUT: if copy then out (Raddr);
     pm (a4)   XVD     NRC ;   if blind  ∨ ready then go to PERHAPS BLIND;
     hv  a3                ;   Raddr:= char;
     ar  b             NTA ;   if -, case free then R:= char 1:= Raddr + case;
     ga  a8            ITA ;   case free:= false;

[ char contains now the next character to be compared. Note that the cha-
  racters in the stack are provided with their case (LC for case indepen-
  dent) and are also increased with e5 (cf. a4)                          ]

     sr  pe9   , pp  p1    ;   R:= R - set marks(stack[p]); p:= p + 1;
     hv  a10           LZ  ;   if R = 0 then goto TEST MARK;

[ The comparison has failed. The following code decides whether
   stack[1:p-i| = stack[i:p-2| concat char, where i = 2, 3, ... p-1      ]

     pm  p     DX          ;   for s:0 2 step 1 until p do begin
a9:  ps  1     , ps  s1    ;     for i:= s step 1 until p do
     ca  s     , hv  a2    ;       if (if i = p then char else stack[i])
     pa  a7    t   e9      ;       ≠ stack[i-s+1] then goto SHORTEN;
a6:  pp  s-1   , pp  p1    ; NEW CHANCE:
a7:  pm  e9    t   1       ;   comment the last p-s+1 characters read
     ca  p1    X           ;   match with stack[1:p-s+1];
a8:  sr [char1]DV          ;
     sr  pe9   V           ;
     pp (a7)   Vt e11 LZ [e11=-e9+1]; p:= p - s + 2; goto NEXT CHAR;
     hh  a6    X       LZ  ; SHORTEN: end;
     hh  a9    X       NZ  ;   go to BEGIN SEARCH;
                           ; TEST MARK:
a10: hv  e7            NA  ;   if NA then go to NEXT CHAR;
                           ; STRING FOUND: comment R = 0;
```

```
           [end actions]
a11: can(b)            NPB ;   out(if -, copy  ∨ case = 0 then
e12: arn 58    DV      LZ  ;     58 else 60);
     [used in pass 1 1a23] ;   if copy then
     pm  60    DX          ;     begin count:= count + 1; go to nextstring end;
a12: hs (e8)               ; IN SECOND STRING:
     pm (b15)  XVt 1   LPB ;   Raddr:= char from drum;
     hh (a)    Dt  1       ;   if Raddr ≥ 0 then OUTR:
     hs  e33           NA  ;     begin out(Raddr);
     hv  a12           NT  ;     go to IN SECOND STRING end;

[ Clearcode appears as -28. -512 marks the end of the string]

     nc  -28   , hh  a     ;   if Raddr ≠ -28 then go to nextstring;
     ps  a12-1 , mt  b11   ;   alter(blindaction) to proceed to: (OUT R);
a13: hv  e45   , ga  a14   ;   Raddr:= 28; go to set summing;
     bs (a4)   t 64e5  NRA ; TREAT SPECIAL: swap; if char > 64e5  ∨ LRA then
     ps  3e7   , hv  e43   ;   begin set return to(NEXT CHAR);
     sc (b1)   DVt -1  NT  ;     treat char end; if R ≥ 0 then begin
a14: pi  0     Vt 573  IZA ; replaced: insum:= insum - Raddr - 1;
     ga  b2    , hv  a4    ;     char:= Raddr; go to TEST end else
     pm (e7)   X       NZB ;   in:= Raddr bits 1 to 3; blind:=LZA:= false;
b48: ac[insum2]DVt 1   NT  ;   if NZB then skip: begin insum2:= insum2 +
     hv  e8    X       LTA ;   last input char + 1; go to NEXT CHAR end;
     hv  e7            NT  ;   if case free then go to OUTPUT;

     hv  e13           LTB ;   if ten then go to trailing edge;
     hv  e8    X       NRB ; ENDMARK: if -, ready then begin blind:= true;
a15: pa  b38   Xt   10.3   ;     go to OUTPUT end; wordend:= 10;
     ps  r1    , hv (e8)   ;   out(Raddr); i:= 239; skip line;
e13: pa  a16   , ud  a15   ; trailing edge: wordend:= 10; i:= 240;
[2]  hv  a18           NQA ;   if -, external then begin
a16: it  -1    , qq (a17)  ;     for i:= i step -1 until 1 do
[-2] pm  10    DX          ;
a17: bt  240   t   -1  IZA ;     outinternal(10); LZA:= false;
     ps  r-3   , hv  e26   ;
<d41,                      ;   for i:= 1 step 1 until filler do
e15: pp  -39   , ps  r     ;     next word out;
[s1] ncn p                 ;
     pp  p-1   , hv  e28   ;
>a18:vy  d43   , pa  a20   ;   select(alarm unit); print:= false; skip line;
e16:                       ; alarmprint: print:= true;
b10: vk  960[=-1.3 see 3e20], it  -1 ;
     vk 1d37+d9, it  -40   ;   fetch last tracks of HELP system;
     lk  -6    , it  -1    ;
     bt  4     , hh  e16   ;
a20: bs 1[print], hv c24   ;   if print then go to ALARM PRINT;
<d41,ncn(e15)  , hs  a23   ;   if filler ≠ 0 then begin only sense:= true; SENSE;
e44: can 1     , hv  a22 > ;     end; if tape output then go to ADJUST TAPE;

[ Termination of the program: ]

a24: arn c1    , hs  c2    ; CALL HELP: fetch catalog
b5:  pi [kind] , arn a     ;   in:= kind; if count ≠ 0 then
     nc  0     , hs  e39   ;     display(count);
<d41,arn b51   , ck  10    ;   R:= if filler ≠ 0 then bits(10,21,paramword)
     tk  -28   , ud  c24   ;     else outtrack;
     ca (e15)  , arn b23   ;   prepare help entry to set twr;
×    arn b23   , ud  c24   ;   M:= first out block;
>    pm  b25   , hv  c74   ;   go to ADJUST SPECIAL;
```

```
<d41,
b6:  gr  i     V           ; see e27
[1b6]qq b6.9+1.19+2574.39  ; buffer base
[2b6]qq b6.9+1.19+2574.39  ; current base
[3b6]qq -d53.9-1.21-2574.39+d53.39              ; form checkword
[4b6]qq 1024.39            ; buffer length

b7:  qqf15.5+15.11+15.17+15.23+15.29+15.35+3.39, ; filemark
[1b7]qq b7.9+1.19          ; filemark to buffer
[2b7]qq 1.19               ; filemark to tape
[3b7]qq  b19   t 1         ; status to buffer, to core

[ Adjust tape: cf. page 13 and Help 3 page 13               ]

a22: vk  d11   , lk  c     ; ADJUST TAPE: select parametertrack;
     vk (c5)   , vk (c4)   ;   fetch search;
     lk  d14   , vk (c4)   ;   fetch selected catalog track;
     pm  b23   , tl  23    ;
     arn(c15)  , ck  8     ;   store[iword]:= store[iword] ∧
     tl  16    , ck  16    ;     8m16016m  ∨ (block count pos 23);
     gr (c15)  , hs  c8    ;   sum track;
     sk  d14   , arn 1b7   ;   write catalog track;
     us        , arn 2b7   ;   paramword:= filemark to tape;
     gr  b52   , us (b35)  ;   write filemark;
     pa  b49   , ps  a24-1 ;   up:= true; only sense:= true;
a23: pa  b55   , hv  e46   ;   SENSE; go to CALL HELP;
>
e
[ Procedure outchar outputs one character to an external unit. The output
  format is defined as follow:
  i. The case for the character SPACE or CARRET is always equal to the
  case of the previous character. (to undefined case in the beginning of
  the tape)
  ii. The procedure does not output superflouos case shifts.
  iii. If the first character (disregarding SPACEs and CAR RETs) is not a
  case shift a LOWER CASE will be output in the front of it.           ]
b a10
e17: nc  58    , ca  60    ; procedure outchar(char); integer char;
     ga  a     , hv  s1    ;   if char = <LC>  ∨ char = <UC> then case 1:= char
     ga  a3    , nc  64    ;   else if char ≠ <CR> ∧ char ≠ <SP>
a:   pm  58[case1]DXV  NZ  ;   ∧ (first  ∨ case 1 ≠ out case) then
[-1] arn a3    , hv  a3    ;   begin if first then begin first:= false;
     ca (a1)   , hv  r-1   ;     outcase:= case 1 end
     bs (a1)   t   60      ;     else
     sc (a1)   D           ;     outcase:= 118 - outcase;
a1:  nt  118   , sy  118   ;     sy(outcase)
     [outcase] [first]     ;     outsum:= outsum + outcase + 1; go to a3
     arn a1    , ud  a4    ;   end else
a3:  sy [char] , ca  61    ; a3: begin sy(char); if char = <SUM> then
b3:  hs  e14 [see 3e37]    ;     treat sum else
     ca  28    , ntn(b14)  ;     outsum:= if char = <CLEAR> then 0
a4:  ac (b14)  DXt 1       ;     else outsum + char + 1
b39: qq [check], hv  s1    ;   end outchar, check is used at 4e46;
e
<d41,
b40: tfault;               ; alarm texts
>
b41: toverlap;             ;
```

[select track]

```
b a10

a:   qq                  ; integer array inbuff[1:7];
     qq
[-3] qq
[-2] qq
[-1] qq 512              ; comment part 1 of 2a -5a are used in Pass 1;
e18: qq                  ; ENDACTIONS:
[6a] qq r-1,             ;   case inbuff[7] of
                         ;     begin
[1] b9: qq IB [=1.39]    ; end track end text: go to end text;
[2]     pa 1e7 t e13     ; end text: begin alter (input next word) to proceed
[3] e19:hv 4i [see e21]  ;  to: (trailing edge); go to end word end;
[4] b11:qq  -1           ; undef 3: go to end word; undef 4:;
[5] b12:qq IZA [= 10.39] ; undef 5: ;
[6] b13:qq DXVN[=960.39] ; end track: ; comment R:= 0;
[7]     hv (e7) Dt -7    ; end word: begin reset pointer; goto NEXT CHAR end;
                         ;     end endactions;
e20: arn -2   , sr b9    ;   if intrack - 1 = last out in pass 1
     sr  b22         NZ  ;      ∨ overlap then
     hv  e29         LZ  ;     go to OVERLAP ERROR;
     hs  e24   , qq  e1-1 ;   select track(intrack); s:= lower buff - 1;
[s1] ac  -2    , pm  b10  ;   intrack := intrack + 1;
     sc  b24   X       ITB;   inlength:= inlength - 1;
     bs (a2)   t   39e1   ;   if wordad ≥ upper buff then
     gs  a2    , ps  39e1 ;     begin wordad:= s; s:= upper buff - 1 end;
     lk  s1    V       NTB;   if inlength ≥ 0 then read track to (s + 1) else
     grn -2    , grn e20  ;     begin intrack:= nonsense; overlap:= true end;
     mb  b11   , is (a2)  ;   cell[wordad + 40]:= cell[wordad + 40] - 1 pos 3;
e21: ac  s40   , hv (e50) ; comment mark end of track; go to end word;
[later changed to:  (e19) ]; comment initially: return;
                         ; NEXT WORD:
e22:a2: pmn 39e1 t 1     ;   wordad:= wordad + 1; M:= cell[wordad]; R:= 0;
<d41,pa  a3    Vt  a-1   ;   skip line;
e23: ps  r-2   , hv  20e1; BUFFERMEDIUM WORD: next word;
     pa  6a    Vt i-6a+2LZ;   if R ≠ 0 then
e32: qq  b40   , hs  e16 ;     alarmprint(↯fault|);
[+2]
×    pa  a3    t   a-1   ; comment no buffer case;
     pa  6a    t i-6a+1  ;
[+1]>                    ;   clear MO;
     cln -6    , ck  -4  ;   for i:= 1 step 1 until 7 do
     ca  63    , sr  b11 ;     begin Raddr:= M ∧ 63; M:= M : 64;
a3:  ga  a-1   t   1     ;     inbuff[i]:= if Raddr = 63 then 64 else Raddr
     hv (6a)             ;     end; go to ENDACTIONS;
e
e24: pm (s1)   , dln b13 ; procedure selecttrack(trackno); integer trackno;
     ar  b13   , ck  20  ;   begin
     gt  r1    , cln -10 ;     vk(trackno : 960 + 960);
[+1] ga  r2    , vk[group];     vk(trackno mod 960)
     pm  b9    X         ;   end;
[+2] vk [track], hr  s1  ;
```

```
b a20
e25: ca  61    , hv  a8    ; outinternal and check: if char = <SUM>
b14: ac[outsum]Dt   1      ;   then go to out and treat sum;
     ca  28    , pa  b14   ;   outsum:= if char = <CLEAR> then 0 else
                           ;     outsum + char + 1;
e26: ca  64    , ar  b11   ; outinternal: if char = 64 then char := 63;
a:   hv  r0    t   2       ;   count:= count + 1; case count of
b17: 0.001                 ;   begin
[2]  ck  10                ;   begin R:= char pos 39;
e27:a1:gr e3-1 Vt 1        ;   oword:= onword + 1; obuff[oword]:= R end;
[buff: gr  b6  V ]         ;   comment buff: oword points to b6, b6:= R;
[4]  ck  16    , ac (a1)   ;   obuff[oword]:= obuff[oword] + char pos 33;
a2:  hv  s1 [used to return]
[6]  ck  -18   , ac (a1)   ;   obuff[oword]:= obuff[oword] + char pos 27;
e35: hv  s1           IQB  ;   comment used at 1e47;
[8]  ck  -12   , ac (a1)   ;   obuff[oword]:= obuff[oword] + char pos 21;
     hv  s1                ;
[10] ck  -6    , ac (a1)   ;   obuff[oword]:= obuff[oword] + char pos 15;
     hv  s1                ;   begin
b38: ar  15.3  D           ;   obuff[oword]:= char pos 9 + wordend pos 3 +
     ac (a1)   , pa  a     ;     set marks (obuff[oword]); count:= 0;
e28: hv  s1 [buff]     NA  ;   if -, NA then begin
     hs  e24           IQB  ;     if buff output then go to next word out;
[s1] pm  b23   , arn b23   ;     LQB:= LB; select(outtrack);
     sr  -2    X       IZA ;     NZC:= outtrack ≠ intrack ∧
     sr  b20           IZB ;           outtrack ≠ inpass;
     arn b9    , ac  b23   ;     outtrack:= outtrack + 1;
a3:  sc  b21   X       ITB ;     outlength:= outlength - 1;
           [see 4a11]      ;
     hv  e29           LTB ;     if outlength < 0 then goto OVERLAP ERROR;
     is (a1)   , sk  s-39  ;     write track from (oword - 39);
     qq (a1)   t  -80  LQB ;     if LQB then oword:= oword - 80;
     hv  s1            NZC ;     if -, NZC then
e29: qq  b41   , hs  e16   ; OVERLAP ERROR: alarmprint(¢overlap¢);
                           ;   end end end; return,
e30:
<d41,arn b50   , ar  b9    ; next word out: rem:= rem + 1;
a4:  gr  b50   V       LT  ;   if rem < 0 then begin
e46: pa  a7    , hh  a5    ;   us(0,b6,rem + current base);
     ar  2b6   , us        ;   return end;
a5:  hv  s1    , pm  b51   ; NEXT BLOCK: rep:= 0;
a6:  bs  1     , hv  a10   ;   if -, first then begin
     arn 3b7   , il (b39)  ; SENSE:
     il        , arn b19   ;   if status error(check) then begin
a7:  bt [rep]  XVt-150 LT  ;     rep:= rep + 1; if 3 < rep then
a10: pa  a6    , hv  a11   ;
     qq  b40   , hs  e16   ;     alarmprint(¢fault|);
     sr  b52   , is (b49)  ;     write(rewrite,paramword -
     bs  s512  , ar  4b6   ;     incr + (if up then 0 else 1024));
b34: us [rewr] , hh  a5    ;     go to SENSE first:= false;
```

```
a11:                        ;
b55: can -1     , hv  s1    ;   if only sense then return;
     arn b51    , ar  3b6   ;   if track = intrack
e31: bs [warn]  , sr  13e1  ;     ∧ warning then
     hv  e29            LZ  ;     go to OVERLAP ERROR;
     arn b9     , ud  a3    ;   outlength:= outlength - 1; if outlength<0
b54: hv  e29            LTB ;   ∧ -,tape output then go to OVERLAP ERROR;
b49: bs  513    , ar  4b6   ;   write(unit,paramword +
b35: us [unit]  , it  512   ;     (if up then 1024 else 0));
     xr (b49)   , ac  b23   ;   block count:= block count + 1;
     bsn(b49)   , ar  4b6   ;   up:= -, up; current base:= buffer base +
     ar  1b6    , gr  2b6   ;     (if up then 1024 else 0);
     arn b52    , ac  b51   ;   paramword:= paramword + incr;
     srn b53    , hv  a4    ;   rem:= -block length - 1; go to next word out;
>                            ; comment end buffer case;
a8:  hs  a                   ; out and treat sum: outinternal(<SUM>);
e14: arn b14    , tk  -5    ; treat sum:
     ar  b14    , ud  e37   ;   char:= outsum : 32 + outsum
     ar  31     D           ;     ∧ 31 + 31;
     pa  b14    , ga  a9    ;   outsum:= 0;
     hr (a)     t   2   LQA ;   if -, external then go to outinternal;
a9:  sy [sum]   , hr  a2    ;   sy(char); return;

e

[ Byte input:
     Bytes output in pass 1 are packed 4 into a word. Last words on even
     tracks are A marked, on odd tracks A and B marked                  ]
b a10
a:   qqf        ,           ; integer array byte [1:4];
     qq         ,           ; comment f is used at END INIT, page 18;
     qq         ,
     qq         ,
e33: a1: pm 39e2 Xt 1        ; byte input: iword:= iword + 1;
     ga  a      , ck  10    ;   R:= set marks (bbuffer[iword]);
     ga  1a     , ck  10    ;   for i:= 1, 2, 3, 4 do
     ga  2a     , ck  10    ;     byte[i]:= part(i) of: (R);
     ga  3a                 ;
b15: pa  3a     Dt   a      ;   reset counter;
     hv  a2             NA  ;   if -, NA then begin byte track: LQB:=LB;
e34: vk  960            IQB ;     wait for track; work length:= work length-1;
b18: btn -1     t   -1      ;     if work length ≥ 0 then begin
     hs  e24                ;       select track(pass input);
[s1] ac  b20    , is (a1)   ;       pass input:= pass input + 1;
     lk  s-39           NZ  ;       read track to(iword - 39) end;
     qq (a1)    t   -80 LQB ;     if LQB then iword:= iword - 80 end;
a2:  arn a      , hv  s1    ;   Raddr:= byte[1]; return;

e
```

```
[     The following procedure is called to examine errorneous input cha-
  racters, checksums or to display the input stream on the typewriter af-
  ter errors.
      The procedure is used in both passes                                ]
b a20
e43: xr        , mb  r1    ; treat char: swap; if (Raddr ∧ 63 ) = 63 then
[1]  ca  63    , hr  s-3   ; ALL HOLES: blind return;
     xr        , it  a9    ;   LT:= true; text:= ↰char↱; skip line;
e36: pa  a     t   a10     ; treat slow: text:= ↰parity↱;
     ca  639   , hr  s-3   ;   if Raddr = 127 + 512 then blind return;
     ga  a4    , gs  a1    ;   char:= Raddr;
     vy  d43               ;   select(alarm unit);
     pm  s-1   X           ;   if cells[s-1] ≠ 0 then
     hv  a2    X       LZ  ; parity or char: begin
     gr  a3    , grn s-1   ;     saved instr:= cell[s-1]; cell[s-1]:= 0;
a:   qq [text] , hs  e38   ; mess: writetext(text);
     bs (b)    , sy  60    ;     if case ≠ 0 then writechar(<UC>);
     pa  a5    t   -511    ;     line count:= 2;
b16:a8:vy [by] , qq        ;     select(saved by);
a1:  ps  -1    , hr  s-3   ;     blind return end;

a2:  hv  a7            LA  ;   if marks(cell[s-1]) ≠ A then
a3:  zq                    ; print: begin execute saved instr:
    [ga        V       NT] ;     if LT then
     hv  a                 ;       go to mess;
a4:  sy [char] , vy (b16)  ;     sy(char); select(saved by);
     ca  64    , it  -1    ;     line count:= line count - 1;
a5:  bt [line] , pm  a3    ;     if line count < 1 then
a6:  gm  s-1           M   ;       cell[s-1]:= saved instr;
     pm  b9    , hr  s1    ;     marks:= 0; return end;

a7:  pm  b26   , ud  a6    ; check: cells[s - 1]:= saveslowaction;
     pm (b1)   DXt  -62    ;   marks[s-1]:= 0; insum:= insum - <SUM> - 1;
     hv  a13           NPA ;   if only input sum then begin
     ga  b14   , nt (b48)  ;     outsum:= insum - insum2;
     qq (b14)  , pa  b48   ;     insum2:= 0 end;
a13: tk  -5    , ar  b1    ;   if (insum : 32 + insum)
e37: mb  31    D[see 1e14] ;     ∧ 31 + 31 ≠ char then
     ar  e37   , nc (a4)   ;
     qq  a11   , hs  e38   ;       writetext(↰tapesum↱);
     hs  b3            LPA ;   if only input sum then treat sum;
     hs  e45           M   ;   if ready then begin A:= true; set summing end;
     pa  b1    , hv  a8    ;   insum:= 0; select(saved by); blind return;

a9: k 63, 58, 29
tchar;
63, 62.
a10:k 63, 58, 29
tparity;
63, 62.
a11:k 58, 29
t tapesum ;
62.

e
```

[bottom of the stack]

```
b a10
e38: pa  a     , it (s)    ; writetext: itext:= part 1(cell[s]) - 1;
a:   pm[itext] t   1       ; next word: itext:= itext + 1; M:= cell[itext];
a1:  arn a2    , cl  -6    ;   comment R(37:39) = 3 1;
     tk  -4    , ga  a3    ; next char: char:= M ∧ 63; M:= M : 64;
     ca  15    , hv  a     ;   if char = 15 then go to next word;
a2:  ca  10    , hv  s1    ;   if char = 10 then return;
     ca  63    , it  1     ;   sy(if char = 63 then 64 else char);
a3:  sy [char] , hh  a1    ;   go to next char;


e39: sy  58    , sy  29    ; display: writered; write LC; writecr;
     sy  64    , tl  -30   ;   M:= Raddr × .001; R:= Raddr : 1000;
     pa  a7    Xt   3      ;   zero:= <SPACE>;
     pt  a5    , mln b17   ;   for i:= 1 step 1 until 4 do
a4:  ck  -10   , ga  a6    ;     begin
a5:  pa  a6    Vt[zero]LZ  ;     if R ≠ 0 then zero := <0>;
     pt  a5    t   16      ;     writechar(if R = 0 then zero else R);
a6:  sy  0                 ;     R:= entier(M × 10);
a7:  bt  [i]   t   -1      ;     M:= M × 10 - R
     mln b12   , hv  a4    ;     end;
     sy  62    , hr  s1    ;   writeblack; return;
e

b a0

[     The following locations overwrite Initialize pass 1.                  ]
d a0=i
<d41,
b50: qq  [remaining.39]    ;
b51: qq                    ; paramword
b52: qq  [increment.39]    ;
b53: qq  [blocklength.39]  ;

b19: qq  >                 ; output status
b20: qq  [pass input.39]   ;
b21: qq  [outlength.39]    ;
b22: qq  [last output track in pass 1.39]
b23: qq  [outtrack.39]     ; [block count.39]
b24: qq  [inlength.39]     ; [work end in pass 1]
b25: qq  [first output block.39]
b26: qq                    ; save slowaction

d e9=i-1, e10=1e9, e11=-e9+1 ; first word in the stack:

[1b26] qq                    ; used only in pass 1
[2b26] qq                    ;        -
d i=a0
e
```

```
b a30
a:   hs  a3               ; table change: index:= next param;
     gi  a2    , hs  a3    ;   type:= next param
a1:  ps  -1    , ps  s1    ;   for s:= 1 step 1 until no of types do
     arn sa7   , nc (a4)   ;     if type = part 1(types[s]) then go to found;
     bsn sa8   , hh  a1    ;   alarmprint(‹param›);
     hv  a5            LZ  ; found:
     tk  10            IRC ;   character table[index]:=
a2:  gr[index] t  e5   MRC ;     types[s] × 2↑10;
     arn a4    , ca  41    ;   if type = replaced then begin
     ps  r1    , hv  a3    ;     part 1[table base + index]:=
a16: pp  p1    , hh  a9    ;     next param; caution:= true end;
     pm  a17   , gm  e47   ;   p:= p + 1; go to take param;
     gi (a2)   , hv  a16   ;
a17: pa  e8    t   e25     ;

a3:  pm  p1    , pp  p1    ; next param: M:= set marks(param[p+1]);
     hv  a5            LC  ;   if marks = end then alarmprint(‹param›);
     cln -30           LA  ;   if marks = underlined then M := M shift -30;
     cln -7    , ck  -3    ;   p:= p + 1; Raddr:= next param:0 M;
     ga  a4    , pi (a4)   ;
a4:  bs  0     t   64      ;   if Raddr > 64 then
a5:  qq  b42   , hs  e16   ;     alarmprint(‹param›)
     hr  s1                ;   return;

[ character types: see also page 1                                          ]

a7:  qqf 37                ;   normal
     qq  49    , qq  -2    ;   alarm
     qq  41                ;   replaced
     qqf 50    , hvf s1    ;   blind
     qq  18    t   512     ;   skip
     qq  53    t   770     ;   end
     qq  51    t   896     ;   case independent
d a8=-i+a7+512
<d41,
a14: arn(c15)  Dt  -1  IZA ; procedure backup; begin comment JJ;
     nc  d14-1 , hv  a15   ;   iword:= iword - 1; if iword < d14 then
     pm (c4)   DXt -1  M   ;     begin track:= track - 1;
     ca  -1    , ac  c5    ;     if track = -1 then begin track:= 959;
     pa  c4    t   959 NB  ;     group:= group - 1; end;
     qq (c8)   t  -2       ;     reltr:= reltr - 2;
     hs  c5                ;     read track;
     pa  c15   t   38d14   ;     iword:= d14 + 38 end;
a15: pm (c15)  , hr  s1    ;   set marks(store[iword]) end backup;
>
[ Start Init pass 1:
     The parameters for the program are examined and the following ac-
  tions are executed:
  i<area>,o<area>: the description of the <area> is searched for and
     stored for later use
  l the control is transferred to the table change routine
  < terminates the scanning. The current input medium is re-selected and
     pass 1 is started                                                    ]
d a12=-126                 ;
e40: qq (c1)   t   2       ; PROGRAM START:
<d35-2, us(-31)t   -96     ;   if aux kind = tape then rewind tape;
>    vk  960   , vk  d11   ;   free is treated as normal area;
     lk  a12   , vk  960   ;   read parameter track;
<d41,pa  c75   t   39a12   ;   assign buffer to read internal;
     pa  c76   t   20a12   ;
     qq (-5)   t   a12-d14 ;
>   [pp  1a12  , ]         ;   p:= base of parameters;
```

```
a9:  pp  1a12  , arn p     ; take params: R:= set marks(param[p]);
     hv  a10           LC  ;   if marks = end then goto end scan;
     hv  a5            NA  ;   if marks ≠ underlined then alarmprint(‹param›);
     ca 35[l]  , hv  a     ;   if R = <l> then go to table change;
     pm  p1    , nc  57 [i];   if R ≠ <i> ∧ R ≠ <o>
     nc 38[o]  , hv  a5    ;      ∨ marks(param[p+1]) ≠ next
     hs  c1    X       NC  ;      ∨ search(p) ≠ 0 then
     hv  a5            NZ  ;       alarmprint(‹param›);
     arn 2c    , ps (p)    ;   s:= Raddr; if s = <i> theN
     bs  s473  , is  58    ;     input area:= areaword else
     gr  sb29  , tl  -7    ;     output area:= areaword; Raddr:= type(areaword);
<d41,pp  p1               ;   if buffermedia then begin p:= p + 1;
     nc  s-35  , hv  a13   ;   if Raddr = tape ∧ s = out then
     hs  a14               ;     begin backup;
[-1] hs  a14               ;     while NA do backup;
     hv  r-1           NA  ;
     hs  a14               ;     backup;
     hs  c15           NA  ;     if NA then get word;
     vk  960   , vk  d11   ;     save search on parametertrack
     sk  c     , vk  d11   ;
     arn 2c    , tl  -7    ;     end end else p:= p + 1;
a13:×pp  p1    , nc  0     ;   if -,buffermedia ∧ 0 < Raddr
     hv  a5            NT  ;      ∨ Raddr = constant
>a6: pp  p1    , nc  -4    ;      ∨ Raddr = ly ∧ s = out
     nc  s-59  , ca  s-39  ;      ∨ Raddr = sy ∧ s = in  ∨ program bit
[-1] hv  a5            LT  ;   then alarmprint(‹param›);
     tk  12    , ud  r-1   ; ignore text: p:= p + 1;
     arn p-1   , tl  -6    ;   if bits 0 to 3 of (param[p-1]) = -1
     ca  -1    , hvn a6    ;   then go to ignore text;
     bs  s473  , pm  p     ;   if s ≠ out then go to take param;
     hh  a9            NB  ;   if marks(param[p]) ≠ number then begin
     pp  p1    , cln -10   ;     p:= p + 1; zerotracks:= param[p] end;
a10: vk  d21   V       LC  ;   go to take params;
     ga  b36   , hh  a9    ;

     lk  d14   , vk  960   ; end scan: fetch catalog track;
     arnd14+d45, gr  1b26  ;   R:= save work as output:= work as output;
     tl  -32               ;
     tln 36    , gt  b33   ;   worklength:= blocks(R);
     tl  1     , nc  0     ;   if worklength > 511 then
     pt  b33   t   511     ;     worklength:= 511;
     tl  -21   , gr  b24   ;   outtrack:= work end:=
     tln 16    , ar  b24   ;     blocks(R) + first block(R);
     gr  b24   , gr  b23   ;
                           ;
     pi  0     , arn -2    ;   if current input ≠ <ly> then
     hh  a11           NC  ;     begin
     pm  -2    , hsn c3    ;     select track(cell[-2]);
<d41,grn b37   , lk  a12   ;     read track to(text buffer);
×    grn b37   , lk  d14   ;
>    pa  e42   t   e41     ;     input:= input internal
a11: vk  3     , mb  a11   ;     end;
     ca  1     , pi  1     ;   twr input:= current input = twr;
     vy  d43-1 t   -d43    ;   select also(alarm unit);
b31: sy  64            LRB ;   if twr input then writecr(2);
     gk  b16   , ud  b31   ;   saveby:= by;
     grn e3-1  t   1   M   ;
     it (r-1)  , bs  78e3  ;   initialize marks in output buffer;
     hv  r-2               ;
     gr  39e3  V       MA  ;   comment see next page;
e                          ;   go to START PASS 1;
```

```
           [test for . ]
b a40                       ;
e41: hs  c27                ; input internal: Raddr:= READ INTERNAL;
[s1] gr  79e3          MC   ;
     hv  a2                 ;
[s3] ca  63    , sr  b11    ;   if Raddr = 63 then Raddr:= 64;
     hr  s1                 ;   return;
e48:
b32: lyn e18   V            ; input external: Raddr:= lyn; skip line;
     arn 10e5  , hv  a28    ; reset and stop pressed: Raddr:= ten; go to TEST END;
     mb  r1    , pm  e18    ;   if (Raddr ∧ 63) = 63 then
[1]  ca  63    , hv  e48    ;     go to input external;
     hr  s1    X            ;   return;

[ Line buffering:
  If the input is taken from the typewriter. generated bytes are buffered
  and output in a burst after every CAR RET. This makes possible the era-
  sing of a line until it is terminated. The action for the line termina-
  tion is started if p < 0 of if the buffer is full. (it can store 80
  bytes)                                                                ]

a:   gr  e1-1  , gs  a2     ; STORE BYTE: car:= car + 1; buffer[car]:= Raddr;
     bs (a)    t 78  e1     ; blind CR: if car > upper bound then
     sy  64    , hv  r3     ;   writecr else
     nc  64                 ;   if Raddr ≠ <CR> ∧
     bs  p1    , hv  s1     ;     p ≥ 0 then return;
[3]  it (a7)   , pt  a3     ;   last state:= state;
     it (a23)  , pt  a4     ;   last case 2:= case 2;
     pp  e1-1  , ps  r      ;   for i:= 1 step 1 until car do
[s1] pp  p1    , arn p      ;     direct out(buffer[i]);
     it  p-1   , bs (a)     ;
     hv (a17)  t   2        ;   comment store end OK and in2nd;
     gi  a5    V            ;   last in:= in; skip line;
                            ;
a1:  qq  b44   , hs  e38    ; ERASELINE: writetext(↯annul↮);
a2:  ps  a11   , pp  0      ; START PASS 1: p:= 0;
     pa  a     t e1  -1     ;   car:= 0;
a3:  pa a7  t0 [last state] ;   state:= last state; comment initially 0;
a4:  pa a23 t [last case 2] ;   case 2:= last case 2;
a5:  pi  256   t   255      ;   in:= last in; comment initially end OK ,-, in2nd;
                            ;   return;
[ Test terminator:
  After every underline (_) regardless to its type, the next character
  will be compared to a period (.). Should the test fail, LQA will be
  set to indicate that the next character must be taken from e18-2.     ]

a6:  hv  s1    , hs (e42)   ; UNDERLINE:
     ps  a11   , ga  e18-2  ;   store char:= input
     pm  14 [_]D       M    ;   if storechar ≠ <.>  ∨ case ≠ 0 then
     ca  59 [.], bs (b)     ;     begin LQA:= true; go to TESTTYPE end;
     hv  a13   X       IQA  ;
```

```
     pm  a34   , arn e18-1 ; STRING TERMINATED:
     qq (b1)   Xt  75  IZA ;   insum:= insum + 59 + 14 + 2; in2nd:= false;
     sy  59            LZ  ;   if slowaction = print then writetext(¢<.¢);
a7:  bt [state]t   -1       ;   state:= state - 1;
     arn(a7)   DXV     IZB ;   if state > -1 then endOK:= state = 0 else
     pan a7    Vt   2  IZA ;     begin state:= 2; in2nd:= true end;
a8:  qq [count]Vt   1  LZB ;   if endOK then count:= count + 1;
     pa  a23   X       LZA ;   if in2nd then case 2:= 0;
     pa  a15   t -e10+1018 ;   byte count:= max allowed;
     hv  a17           NZB ;   if -, endOK  ∨ -, twr input then begin
     hv  a17           NRB ;     outbyte(512); go to NEXT CHAR end;
     pm  a8    X           ; TRIPLE TERMINATION: writecr; p:= -1;
     pp  -1    , hs  e39   ;   display (count);
     sy  64    , sy  64    ;   writecr(2);
     pm  e18-1 , hv  a16   ;   out M (512); go to NEXT CHAR;

a9:  pi  0     Xt  -9  LQA ; CASE SHIFT:
a10: bt[shifts]Xt -150 LRB ;   shifts:= shifts + 1; if shifts > 3
     hvn a24   X       LRB ;     ∧ twr input then go to NUMERIC;
d a11=i-1                  ; NEXT CHAR:
     hs (a13)          LPC ;   if blind then execute(action[char + table base]);
e42: [-3] hs e48       NQA ;   char:= Raddr:= if -, LQA then input
     pm  e18-2 , ud  a9    ;     else storechar; LQA:= false;
a34: gaf a13   V       NT  ;   if parity  ∨ slowaction ≠ 0 then
       [ f is used at 1e47];     begin only input sum:= false;
a12: hs  e36   t   a11 IPA ;     treat slow end;
     nc  64 [CR]       LRB ;   if Raddr = <CR> ∧ twr input then
     ca  14 [_]V       IPC ;     p:= negative; blind:= false;
     pp  a-1a11, hv  a13   ;   if Raddr = <_> then begin
     hh  a6            IZB ;     endOK:= false; go to UNDERLINE end;
     ac (b1)   Dt  1   LQB ;   if sum then insum:= insum + Raddr + 1;
     nc  13            LRB ;   if Raddr = <aa> ∧ twr input then
a13: pm [char] Vt  e5  IPC ;     begin writecr; go to PERHAPS ERASE end;
     sy  64    , hv  a27   ; TEST TYPE: M:= set PC(character table[char]);
     nc  58    , ca  60    ;   if Raddr = <LC>  ∨ Raddr = <UC> then
     hv  a10               ;     go to CASE SHIFT;
     pa  a10   , ca  28    ;   shifts:= 0; if Raddr = <CLEAR>
     mt  -1    D       LZA ;     ∧ in2nd then Raddr:= -Raddr;
     hv  a22   X       NPB ;   if spec then begin swap; go to TREAT SPECIAL end;
a14: hv  a29           LZA ; after spec: if in2nd then go to CHECK CASE 2;
     hh  p1a11         LPC ;   if blind then go to if p < 0 then blind CR
     pi  0     Xt  -257    ;     else NEXT CHAR; endOK:= false;
a15: ncn[bytes]XVt  -1     ; next out: byte count:= byte count - 1;
a16: ps  a11   XV          ;   if byte count = 0 then alarmprint(¢full¢);
     ar  b     , ud  a19   ;   outbyte(Raddr + case + character table base);
     hv  a20           NZC ;   go to NEXTHCAR;
                           ; out M: set return to (NEXTHCAR); swap;
```

```
a17: hv  r0    t   2   NRB ; OUTBYTE: if twr input then go to STORE BYTE;
     hv (a)    Dt  1       ; directout: pos:= pos + 1; case pos of begin
[2] a18: gr e3-1 [oword]t 1;     begin oword:= oword + 1;
     hv  s1                ;     obuff[oword]:= Raddr end;
[4]  ck  -10   , hh  r2    ;   obuff[oword]:= obuff[oword] + Raddr pos 19;
a19: ar  e5    DV          ;
       [used at 2a15]      ;
[6]  ck  -20   , ac (a18)  ;   obuff[oword]:= obuff[oword] + Raddr pos 29;
     hv  s1                ;     begin
[8]  ck  10    , ac (a18)  ;     obuff[oword]:= set marks(obuff[oword])
b30: arn -1    DVt  1  LA  ;     + Raddr pos 39; pos:= 0; if LA then
       [used tracks]       ;       begin used tracks:= used tracks + 1;
a33: pa  a17   , hv  s1    ;
b33: gm  2b26  , ca[length];       if used tracks := worklength then
a20: qq  b43   , hs  e16   ;         alarmprint(⊀full⊁);
     arn b9    , sc  b23   ;       outtrack:= outtrack - 1;
     hs  e24               ;       select track(outtrack);
     qq  b23   , pm (a18)  ;
     is (a18)  , sks -39   ;       writetrack from (oword - 39);
     qq (a18)  t -80   LB  ;       if Bmark(obuff[oword]) then
     pm  2b26  , hv  a33   ;         oword:= oword - 80;
                           ;   end end end; return;
                           ; TREAT SPECIAL:
a22: bs (a13)  t 64e5  NPA ;   if char > 64  ∨ -, NPA then
     ps  a12   , hv  e43   ;     begin set return to(NEXT CHAR);
     ca  512   , hhp 1a11  ;     treat char end; if skip then go to if p < 0
     nc  896   XV      LT  ;     then blind CR else NEXT CHAR;
     ga  a13   , hv  a13   ;   if replaced then begin char:= Raddr;
     hv  a28   X           ;   go to TEST TYPE end; if -,case free then
     sr  b     XV      NZA ;   go to TESTEND; if -,in2nd then begin endOK:=false;
                           ;     Raddr:= Raddr - case; go to next out end;
a29: pm  b     XV          ; CHECK CASE 2:
     hv  a15           IZB ;   if case ≠ case 2 then
a23: ca[case2] , hv  a16   ;     begin
     ga  a23   , ud  e12   ;     case 2:= case; outbyte(if case = 0
     arn 60[UC]D           ;     then 58 else 60)
     ps  a16-1 , hv  a17   ;     end; go to outbyte;

a32: qq  b47   , hs  e16   ; SYNTAX: alarmprint(⊀termination⊁);

a24: ca  60    , hv  1a11  ; NUMERIC: if char = <UC> then go to NEXTCHAR;
     syn 29    , sy 17 [<] ;   case:= numb:= digit:= 0; writered; writechar(17);
a25: pa  b     , ml  b12   ; DIGIT: numb:= 10 × numb + digit;
[1a25] lyn a26 , ca  60    ; INCH: R:= digit:= lyn; if Raddr = <UC> then
     pan b     t   128     ;     begin R:= 0; case:= 128 end;
     ca  58    , pan b     ;   if R = <LC> then R:= case:= 0;
     hv  1a25          LZ  ;   if R = 0 then go to INCH;
```

```
     ca  16[0] , pan a26   ;   if R = 16 then digit:= 0;
     is (b)    , bs  s511  ;   if case = 0
a26: bs [digit]t   9       ;     ∧ digit ≤ 9 then go to DIGIT;
     cln -10   V           ;   shifts:= 0;
     ck  10    , hh  a25   ;   char:= numb;
     ga  a13   , pa  a10   ;   writeblack;
     sy  62    , hv  a13   ;   go to TEST TYPE;
                           ; PERHAPS ERASE:
a27: hv  a1            NZB ;   if -, endOK then go to ERASE LINE;
a28: pm  513   DXV     LZB ; TEST END: if -, endOK then
     ca  832   , hv  a32   ;   go to if Raddr = ten then SYNTAX
     hv  a14   X       NZB ;     else after spec;
     pp  -1    , hs  a17   ;   p:= negative; outbyte(513);
[s1] bt  159   t   -1      ;   for i:= 159 step -1 until 1 do
     hvn(a17)  t   2       ;     direct out(0);
```

[The tracks containing the bytes were output from the end of the working
  area backward. The order of the tracks will be reversed, thus during
  pass 2 the tracks will become free for output as the bytes are processed.]

```
     arn b23   , gr  b20   ;   pass input:= outtrack;
     gr  b22               ;   last track:= outtrack;
     pp (b30)  , pp  p1    ;   p:= work length:= used tracks + 1;
a30: gp  b18   , bs  p510  ; test middle: if p < 2 then
     vk  960   , hh  a31   ;      go to reversed;
     pp  p-2   , hs  e24   ;   p:= p - 2; select track(outtrack);
     qq  b23   , lk  e1    ;   read track(buf 1);
     sc  b24   , hs  e24   ;   workend:= workend - 1; select track(workend);
     qq  b24   , lk  40e1  ;   read track(buf 2);
     sk  e1    , hs  e24   ;   writetrack(buf 1); select track(outtrack);
     ac  b23   , sk  40e1  ;   outtrack:= outtrack + 1; write track(buf 2);
a31: hh  a30   , vk  c63   ;   go to test middle;
     lk  e1    , lk  40e1  ; reversed: fetch INIT MEDIUM;
     arn a8    , nc  0     ;   if count ≠ 0 ∧ -, twr input then
     hs  e39           NRB ;     display(count);

e                          ; end Pass 1;
```

```
[ Select and initialise input and output media for Pass 2:                ]
b a30
d a = e1 + c82 - c28       ; define bufferpart

     arn b28   , ga  b5    ;   R:= output area; kind:= Raddr;
     tl  -2    , pa  b16   ;   saved by:= 0; if Raddr : 4 = work then
     ca1.5+d32.7,it  b46   ;     R:= work as output;
     arn b28   , tl  -7    ;
     hv  a1            NT  ;   if kind = sy then begin
     tk  17    , ga  b16   ;     saved by:= part 2(R);
     pa  e8    t   e17     ;     external:= true; out:= outchar;
     pa  a5    , hvn a12   ;     go to INIT INPUT end;
a1:<-d41+1,                ;   if -,buffermedia then begin
     tl  -25   , tln 16    ;     outlength:= blocks(R);
     gr  b21   , tln 16    ;     outtrack:= firt output block:=
     gr  b25   , gr  b23   ;     first block(R);
a12: qq  e2    , vk  960   ;     go to INIT INPUT end;
×    qq  e2    , ga  a2    ;   INIT MEDIUM(R);
     tl  7     , hs  e1    ;   outlength:= length in init;
     pm  e1    , gm  b21   ;   go to case kind + 1 of
a2:  hhr [kind], arn -2    ;  (DRUM,
[1]  hh  a4    , pt  a11   ;   DISC,
[2]  hv  a3    , pm  4b6   ;   CARR,
[3]  hv  a9    , pm  a13   ;   TAPE);
```

```
     hv  a14          NZ  ; TAPE: if R ≠ 0 then go to label error;
     arn b28   , cm  b27  ;   if unitpart(input area) ) unitpart(output area)
     pa  b34   Vt  32     ;     ∧ kind(input area) = tape then
     qq  b41   , hs  e16  ;     alarmprint(↓overlap|); rewrite:= 32;
     pa  e44   , grn b23  ;   type output:= true; block count:= 0;
     grn b54   , hv  a10  ;   go to COMMON;
a3:  arn b28   , sr  b27  ; DISC: disc:= true; if input area = output area
     pa  e31   t   1   LZ ;     then warning:= true; go to COMMON;
a4:  hv  a10   , gr  b25  ; DRUM: outtrack:= first output block:=
     gr  b23   , pa  e15  ;   track in init; filler:= 0;
     hvn a12              ;   go to INIT INPUT;
a9:  arn 2     DV         ; CARR:= M:= 1024; R:= 2 pos 9; skip line;
a10: pm  3a    , arn 1a   ; COMMON: M:= block length in init; R:= incr in init;
     gr  b52   , gm  b53  ;   block length:= M; increment:= R;
     srn b53   , sr  b9   ;   rem:= -block length - 1;
     gr  b50   , ar  2a   ;   paramword:= current block in init + incr in init
     ar  1a    , ar  a16  ;     + rem + 2568;
     gr  b51   , arn b6   ;
     gr  e27   , arn a15  ;   buffer output:= true;
     gr  e28   , arn 4a   ;   check:= check in init;
     ga  b39   , tk  10   ;   unit:= unit in init +
a11: ar  16    Dt  -16    ;     (if disc then 16 else 0);
     ga  b35   , it (b35) ;   rewrite:= rewrite + unit;
     qq (b34)  , cln -10  ;   filler:= block length - 39;
a12: vk  e2    , ac  e15  ;
>    arn b27   , hs 40e1  ; INIT INPUT: if INIT MEDIUM(input area) ≠ 0
<d41,hv  a14          NZ  ;   then go to label error;
>    pm  -2           IQC ;   if current input = external then
     hh  a8    X      NC  ;     go to EXTERNAL INPUT;
     pm  40e1  , gm  b24  ;   inlength:= length in init;
     pp  40e1  , hv  e20  ; init inchar internal: end track;
e50: hv  r0    Vt  e49 NQB ;   if current input ≠ drum then begin
<d41,vk  960              ;     wait for track;
     pa  18e1  t   11e1   ;     get state; comment for comparsion
     arn 18e1  , il       ;     with output block;
     pa  1e7   t   e23    ;   input:= BUFFERMEDIUM WORD end else
>[-1]pt  e21   Vt  e19    ;
d e49= i-e50-1            ;
[1]  gp  e22   , hv  e20  ;     begin wordad:= upperbuf; end track end;
a17: vy (b16)  , pm  r    ; END INIT: select(saved by); LB:= false;
     pp  79e2  , hs  e34  ;   byte track; LB:= true;
     gp  e33   , hs  e34  ;   iword:= e2 + 79; byte track;
b36: a7: bt [zero] t -1   ;   for i:= 1 step 1 until zerotracks do
     pa  a6    Vt   240   ;     for j:= 1 step 1 until 240 do output(0);
a5:  pi 8 [LQA], hv  e6   ;   go to START PASS 2;
a6:  bt  240   t   -1     ;
     ps  r-2   , hvn(e8)  ;
a8:  hv  a7    , vy  1    ; EXTERNAL INPUT: select(twr input);
     ac (b16)  D          ;   saved by:= saved by + current input;
     pp  e13   , ca  d17  ;   if current input = reader ∧ NRB ∧
b37: lyn       D      NRB ;     -, internal input in Pass 1 then lyn;
     gp  1e7          MA  ;   place external input instructions;
     pm  b32   , gm  e7   ;   go to END INIT;
     hv  a17              ;
```

```
<d41,
a13: qq -1.2+1.23-1.27     ; mask kind ∧ unit
a14: qq  b45   , hs  e16   ; label error: alarmprint(¢label|);
a15: hv  e30               ; buffer output
a16: qq 2568.39            ; 1543 + 1024 + 1
>
b27: qq -1.2+d17.19        ; input areaword: reader
b28: qq -2.2+32.19         ; output areaword: punch

d b29=b27-57, b46=1b26-b28 ;

b42: tparam;               ; alarm texts
b43: tfull;                ;
b47: ttermination;         ;
b44: k 29
     tannul;
63, 62.                    ;
<d41,                      ;
b45: tlabel;               ;
>
e                          ; end init pass 2

< i-c27+10, ilength
>
< i-c48   , ilength1
>
```

```
d i=i+39, d=k-d1           ; d = no of tracks
b k=d42, i=0, a10          ;
d a1=d19-960               ; a1= group no for image
d a=d, <d35, a=2 >         ; a = no of blocks

<d39,                      ; if aux only then
d i=d2                     ;
     hs  1                 ;   begin
     hv  a5                ;
<d36,                      ;
     tres;                 ;   (if aux reserved then
     qqf a.39              ;   res, no of blocks,
x<d39,                     ;   else
     tset;                 ;   set, aux kind, no of blocks,
     qqf d35.39            ;   typein)
     qqf a.39              ;
[STOP, SUM]aibase, edit
s
[STOP, CLEAR]
><d39,                     ;
     qq  39,               ;   concat p i d 0,
     qq  57,               ;
     qq  52,               ;
     qqf                   ;
     tedit;                ;   edit, spec <
     qqf d.9+e40.19+e4.29  ;
     qqf,                  ;

a5:  hs  1                 ;
     hv  a6                ;
     tmove;                ;   move, b load place, edit <
     qq  50,               ;
     qqf d.23+d1.39+a1.29-a1.33      ;
     tedit;                ;
     qqf,                  ;

a6:  hs  1                 ;
     hv  a7                ;
     tsetsum;              ;   setsum, edit<
     tedit;                ;
     qqf,                  ;
d d2=i                     ; end else
a7:  hsf 2                 ;
x                          ;
d i=d48                    ;
     qq d35.2+11.7+d36.5+d.23+d1.39, ; load to primitive catalog;
     qq,                   ;
     tedit;                ;
     qq d.9+e40.19+e4.29   ;
d d48=i                    ;
     qqf                   ;
>                          ;
d d1=d+d1                  ;
e                          ;
e                          ; end of outermost block
[STOP, SUM]a  iedit
s
```

[STOP, CLEAR]

```
b k=d1, i=40d13, a10       ;   begin exit
a=80d13                    ;   begine base of image track 0
     hv  a1    , vk  d16   ; 0 ENTRY FROM PROGRAMCALL: goto bypass;
a2h: sk  a     , it  1     ; 1 READ IMAGE: theese instructions are
     vk  d16   , it  40    ; 2 executed in cell 0-6; write image track 0;
     lk  0     , it  -1    ; 3 read image track 1-24;
     bt  23    , hh  ra2   ; 4 read image track 0;
     vk  d16   , lk  0     ; 5 restore group and wait for drum;
a1:  vk  d19   , vk  d16   ; bypass: read image track 0;
     [this instruction will usually restore group. In step execution
      group will be restored in the instruction on image track 0.]
     lk  a     , vy  512   ;   inhibit by;
     vk  25d16 , ps  9     ;   init cells:= 0;
     pm  e     , arn 8a    ;
     cm  e14           NC  ;   top init:= 9;
     pp  9     , hh  a3    ;   if core[8] ≠ vy 1 t 511  ∨
     pm  a4    , arn 9a    ;     core[9] ≠ qq , hv/hh (rx) then
     cm  9d12          LA  ;     goto init;
     pp  9     , hh  a3    ;
     ps  6     , pp  7     ;   init cells:= 6; top init:= 7;
a3h: gt  9d12  , ps  s-1   ;   std(10-19):= core[9] (10-19);
     pm  pd12  t  -1   IRC ; init: for i:= top init - 1 step -1 until
     gm  pa    t  -1   MRC ;   top init - init cells do
     bs  s     , hh  a3    ;   core[i]:= std[i];

     pm  9a    X       IRB ;   s:= 0; RB:= marks(core[9]);
     ga  9d12  , arn  1d13 ;   std(0-9]:=core[0](0-9); if no params then
     hvn a5            LC  ;     begin Raddr:= 0; goto set core 9 end;
     pa  9d12  , ca  56    ;   std(0-9):= 0; if param 1 = h then
     arn 2d13  , is  1     ;   begin Radr:= param 2; RC:= 11 end else
     pi  s2    , tk  30    ;   begin Radr:= param 1; RC:= 10 end;
     hv  c58           LC  ;   if not number then
     hv  c58           NB  ;   goto param alarm;
a5:  lk  1000  , ar  9d12  ; set core 9: read image track 25 and registers;
                           ;   Raddr:= Raddr + std(0-9); core[9]:= if RB then
     ar  7d12          LRB ;   qqf Radr, hh (rx) else
     gr  9a            MRC ;   qq Radr, hv (rx);
     vk (c5)   , vk (c4)   ;   select(exit track);
     ps (7)    , arn 1     ;   restore s;
     ga  6a    , arn 2     ;   prepare restore group;
     vy (2)    t 512       ;   restore by 1-9;
     gt  6a    , arn 11    ;   prepare restore track;
     pm  8             IRC ;   restore R00-39; RA:= 0; RB:= overflow;
     tl  39    , pm  6     ;   restore M and marks; restore p;
     pp (12)   , hv  c59   ;   goto set O; [page 6 of help]

a4:  qq -1.39+1.19-1.33    ;   mask for checking cell 9 of image
a:                         ;   exit length
```

```
i=39i, d=k-d1                ;
<d35, iexit kind;            ;   if aux kind > drum then alarm message;
s                            ;
>                            ;
b k=d42, i=0, a10            ;   load to image
a1=d19-960                   ;
a=d                          ;   a = no of blocks
i=d2                         ;
<d39                         ;   if aux only then
a4:  hs  1                   ;     begin
     hv  a5                  ;     move, b loadplace, exit <
     tmove;                  ;
     qq  50    ,             ;
     qqf d.23+d1.39+a1.29-a1.33        ;
     texit;                  ;
     qqf       ,             ;

a5:  hs  1                   ;     setsum, exit <
     hv  a6                  ;
     tsetsum;                ;
     texit;                  ;
     qqf       ,             ;
d2=i                         ;
a6:  hsf 2                   ;
×                            ;     end else
i=d48                        ;
     qq 11.7+d36.5+d.23+d1.39          ;
     qq                      ;
     texit;                  ;
d48=i                        ;
     qqf                     ;     load to primitive catalog;
>                            ;
d1=d1+d                      ;
e                            ;   end load to image
e                            ;   end exit
[STOP, SUM]ai exit
s
```

```
b k=d1, i=40d13, a50, b50, e40

d e11=0, e18=0              ;
<d41,<d53-399,<-d53+401,    ; e18= disc with 400 words blocks available
d e18=1 >                   ;
<d41,e11=-e18+1 >           ; e11= other disc availabe
d e10=c74-400               ; core buffer base

[40d13] [input areaword]
b47: pp  d13-1 , hs  e19    ;   p:= param base;
b4:  tk  10    , gt  b12    ;   take input param;
<d41,
b38: hv [ii]   , qq  >      ;   [init input action]
[s3] [output areaword]
b37: hs  e19                ;   take output param;
b5:  tk  30    , ga  b13    ;
<d41,
b39: qq        , hv [io] >  ;   [init output action]
[s3]
<d35-2,us(-31) t   -96   >  ;   if aux kind = tape then rewind tape;
a43: pp  p1    , pm  p1     ;   if kind (param[p+1]) ≠ end then
     hv  a45           LC   ;     begin
a46: mln b8    XV      LB   ;     if kind(param[p+1]) ≠ numb  ∨
a37: qq  b44   , hs  c24    ;     kind (param[p+2]) ≠ end  ∨
     sc  b4            ITA  ;     40 × param[p+1] > inlength then
     hv  a40           LTA  ;     alarmprint(¢length¢) else
     gr  b4    , grn a46   ;     inlength:= 40 × param[p+1]
<d41,hv  a43             >  ;   end;
a45: arn b37   , vk  960    ;   fetch(INIT MEDIUM);
     vk  c63   , lk  d14    ;
     qq  e10   , vk  960    ;
     pa  3d14  , hs  d14    ;   if INIT MEDIUM (out areaword) ≠ 0 then
<d41,hv  a42           NZ   ;   alarmprint(¢label|);
     hv  b39                ;   go to init output[kind(out areaword)];
>
e20: arn 1c    , gr  b1     ; init drum output:
<-d41+1,
     gr  b14   , vk  c63    ;   outbase:= track; go to INIT INPUT;
×    gr  b14   , hv  a32    ;
d a31 = c82-c28+d14
e21:                        ; init disc output:
<e18,arn 2a31  , ar  1a31   ;   if e18 then begin
     ar  b45   , gr  b24    ;   outbase:= cur block + 1;
     pp (4a31) , it  p      ;   check:= check block;
     pt  b21   , it  p      ;   unit:= check block;
     pt  b22   , hv  a32    ;   go to INIT INPUT end
><e11,                      ;   else begin
     arn a17   , ar  b10    ;   block length:= d53;
     gr  b26   , arn b25    ;   incr:= 1 pos 21; paramword:=
     gr  b27   , ar  2a31   ;     incr + cur block + 1550;
     ar  b45   , gr  b28    ;
     it (4a31) , pt  b29    ;   unit:= check:=
     it (4a31) , pt  b30    ;     check block;
     hv  a32                ;   ro to INIT INPUT end;
><d41,
e22: arn 2a31  , ar1 a31    ; init carr output:
     ga  b28   , hv  a32    ;   reel and block:= cur block + 1; go to INIT INPUT;

e23: arn 4a31  , gt  b18    ; init tape output:
     ck  -10   , gt  b17    ;   check:= check block; unit:= read block;
     grn b31                ;
     pm  d13   , gm  c      ;   save search on parametertrack;
     vk  960   , vk  d11    ;
     sk  c     , vk  d11    ;
```

```
a32:<d41, vk  c63  >       ; INIT INPUT:
     lk  d14   , vk  960   ;   fetch(INIT MEDIUM);
     qq  e10   , pa  3d14  ;   medium:= false;
     arn b47   , hs  2d14  ;   INIT MEDIUM(input areaword);
<d41,hv  a42          NZ   ; if R ≠ 0 then alarmprint(¢label|);
     pm  b37   X          ;   Raddr:= kind(out areaword);
     tl  -7    , hv  b38   ;   go to init input[kind(input areaword)];
>
e24: pm  1c    , gm  b0    ; init drum input: inbase:= track;
<d41,nc  0     , hv  a33   ;   if Raddr = drum ∧
>    arn b1    , sr  b0    ;     inbase < outbase then
     hv  a33          LT   ;     begin
     pm  b7    , gm  b6    ;     reverse:= true;
     grn b7    , pm  b4    ;     backward:= 1; forward:= 0;
     dln b8    , ac  b0    ;     R:= inlength : 40; inbase:= inbase + R;
     ac  b1    , pm  b1    ;     outbase:= other end:= outbase + R;
     gm  b14          MB   ;     end;
<d41,hv  a33              ;   go to MOVE ON;

e25: pm  2a31  X          ; init disc input: inbase:= cur block + 1;
     ar  b25   , gr  b23  ;
     xr        , ca  1    ;   if Raddr = disc ∧
     xr        , sr  b24  ;     inbase < outbase ∧
     ar  b45   , pm  b4   ;     outbase <
     hv  a34          NT  ;     inbase + inlength : block length
     dln b10   , tk  18   ;   then alarmprint(¢overlap|);
     ar  b23   , sr  b24  ;
     hv  a41          NT  ;
>a34:                     ;   if e18 then begin
<e18,pp (4a31) , it  p     ;   check:= check block;
     pt  b19   , it  p-16  ;   unit:= check block - 16;
     pt  b20   , arn b23   ;   transfer(check-16,inbase);
     il  p-16  , hv  a33   ;   go to MOVE ON end;
><d41,
e26: pa  c66   t   c-2     ; init carr input:
     pa  c67   t   c-3     ;   prepare get word;
     pa  c68   t   c-1     ;
     pa  c73   t   c-1     ;   goto MOVE ON;
     hv  a33               ;
e27: nc  3     , hv  a35   ; init tape input:
     arn b17   , ck  10    ;   if Raddr = tape ∧
     nc(4a31)  , hv  a35   ;     output unit = input unit then
a41: qq  b43   , hs  c24   ;     alarmprint(¢overlap|);
a35: arn 4a31  , gt  b16   ;   check:= check block;
     ck  -10   , gt  b15   ;   unit:= read block;
     arn b48   , ud  b16   ;   transfer(param);
>
a33:
```

```
     arn b5    , sr  b4    ; MOVE ON: if outlength < inlength ∧ -, tape output then
b31: hv  a40           LT  ;   alarmprint(‡length‡);
     srn b4    , pp  r     ; next action: input;
     hv  a40           LZ  ;   if inlength = 0 then alarmprint(‡length‡);
     gr  b4    , gr  b5    ;   inlength:=outlength:= -inlength;
     pi  0     , hv  e13   ;   LZA:= false; go to SWITCH STATE

[take input param, take output param:]

e19: gs  a36   , hs  c52   ; take param:
     hv  a37   , nc  50[b] ; number: go to ERROR; single: if R = <b> then skipline;
     hv  a37   , hv  a38   ; end list: go to ERROR; text: go to store word;
     pp  p1    , arn p1    ;   p:= p + 1; R:= param[p+1];
     hv  a37           LC  ;   if kind(R) ≠ numb then
     hv  a37           NB  ; ERROR: alarmprint(‡param‡);
a38: hv  a37           LT  ; store word: if kind(R) < 0 then go to ERROR;
     ga  b46   , tl  -2    ;   kind:= part 1(R);
     ca 1.5+d32.7 t 512    ;   if R = work area ∧ take output then
     arnd14+d45, hh  r1    ;     R:= work as output;
     tl  2     , gr (a36)  ;
<d41,tl  -7                ;   cell[s]:= R;
     ga  a39   , tl  -25   ;   M:= blocks(R);
a36: ps [s]    , tln -23   ;   cell[s+2]:= set marks(medium table
a39: arn[kind] t   e30     ;     [kind(R)]);
     gr (s2)   D           ;   store input or output action;
     ud  s1    , ud  s1    ;   M:= M mult
     mln b8 [40]       NC  ;     (if NA ∧ NB then 40 else
     mln b9 [400]      LC  ;     if LA ∧ LB then 400 else
     mln b10[d53]      LA  ;     if LA ∧ NB then disc block length else
     tl  9  [512]      LB  ;     if NA ∧ LB then 512 else not possible);
     gm  s1    , hv  s3    ;
×a36:ps [s]    , tl  -7    ;   cell[s+1]:= M;
     nc  0     , hv  a37   ;   return;
     tl  -25   , tln -23   ;
     gm  s1    , hv  s2    ;
>
[constants:]

<d41,
b8:  qq  40.39             ; track length
b9:  qq 400.39             ; block length, tape
b10: qq d53.39             ;   -     -   , disc
b25: qq   1.21             ; block increment, disc
b45: qq1550.39             ; input buffer length


b41: tfault;               ;
b42: tlabel;               ;
b43: toverlap;             ;

a42: qq  b42   , hs  c24   ;
×b40:
>
b44: tparam;               ;
b49: tlength;              ;

a40: qq  b49   , hs  c24   ;
```

```
[drum input description]

b0:  qq [input base.39]
[p-1]lk (b2)   t   40      ; input instruction
e0:  qq  b4    , hv  e13   ; input length, endaction: SWITCH STATE;
[p1] hvf a0                ; next action: OUTPUT [fmark]

[drum output description]

b1:  qq [output base.39]
[p-1]sk (b2)   t   40      ; output instruction
e1:  qq  b5    , hv  e14   ; outlength, endaction: EXIT
[p1]                       ; next action: INPUT [no f mark]

[drum I-O driver:]
                           ; DRUM:
a0:  pa  a1    t   9       ;   for i:= 1 step 1 until 10 do begin
<d41,pa  b2    t e10-40    ;     iword:= buf base - 40;
×b2: pa[iword]Dt e10-40    ;
>
a2:  arn b6    , sc  p-2   ;     base:= base - backward;
     pmn p-2   , hs  c3    ;     else begin select track(base);
     arn b7    , ac  p-2   ;     base:= base + forward; iword:= inword + 40;
     arn(p)    , ud  p-1   ;     execute read or write end;
     ar  b8    , gr (p)    ;     R:= length:= length + 40;
a1:  bt [i]    Vt  -1  LT  ;     if length ≥ 0 then go to wrapup drum;
<d41,tk  30    , hv  a4    ;     end;
×    hv  p                 ;
>    hv  a2                ;
e13:b12: pm p1 , pp e0 [in]; SWITCH STATE: p:= if Bmark(cell[p])
b13: pp  e1 [out]      LB  ;     then output descr else input descr;
     vk  0     , hv  p1    ;   go to action[p];
<d41,
a4:b2: qq [iword] t 40     ; wrapup drum: iword:= iword + 40;
[1b2]sc  b2    , ps (b2)   ; wrapup: for j:= iword - R step 1 until
[-2] can s-400e10 , hvp    ;     buf top do buf[j]:= 0;
     grn s             M   ;   go to end action[p];
     ps  s1    , hv  r-2 > ;
b6:  qq                    ;   integer backward,
b7:  qq 1.39               ;           forward;
<-d41+1,b8:>
b11: qq 1.39               ;   track length no buffermedium case
b14: qq[first out block.39];   also block count

e14: srn r     , hs  c2    ; EXIT: get catalog track;
     arn b1    , pm  b14   ;   R:= output base; M:= set marks (other end);
     qq        X       LB  ;   if LB then swap;
b46: pi [kind] , hv  c74   ;   in := out kind; go to ADJUST SPECIAL;
```

```
[tape input description]
<d41,qq e10.9+400.19+1550.39
b48: qq 400.19+1550.39
b15: qq  b4    ,   il      ; input length, check
b16: qq  b6    ,   il      ; increment = 0, transfer B < -T
e2:  hv  e13   , il  0     ; endaction: SWITCH STATE; transfer F < -B
[p1] qqf 0     , hvf a5    ; next action: OUTPUT

[tape output description]
[p-4] qq e10.9+400.19+3100.39
     qqf 400.19+3100.39
b17: qq  b5    ,   il      ; out length, check
b18: qq  b6    ,   ps      ; increment = 0, transfer T < -B
e3:  hv  a23   , us  0     ; endaction: TERMINATE TAPE; transfer B < -F
[p1] qq  -1    , arn b7    ;   block count:= block count + 1;
     ac  b14   , hv  a5    ;
>
[disc input description]
<e18,qq e10.9+400.19+1550.39
b23: qq 400.9 [+input base.21] + 1550.39
b19: qq b4     ,   il      ; input length, check
b20: qq  b25   ,   il      ; increment, transfer B < -D
e4:  hv  e13   , il  0     ; endaction: SWITCH STATE; transfer F <- B
[p1] qqf 0     , hvf a5    ; next action: OUTPUT

[disc output description]
     qq e10.9+400.19+3100.39
b24: qq 400.9 [+output base.21] + 3100.39
b21: qq  b5    ,   il      ; outlength, check
b22: qq  b25   ,   us      ; increment, transfer D < -B
e5:  hv  a28   , us  0     ; endaction: TERMINATEDISC; transfer B < -F
[p1]
>
[disc-tape-I-O driver]
<d41,
a5:  qq  -1    V           ; SENSE: only sense:= true;
e12: ps  s1    , gs  e15   ;
     pm  p1    X           ; DISC: TAPE:
     nc  0     , hv  a8    ;   if first out then begin first out:= false;
e16: pa  a6    , hv  a7    ;   go to TEST; rep:= 0; go to SENSE;
a6:  bt  0     t   -150    ; ERROR: rep:= rep + 1; if 3 < rep then
a26: qq  b41   , hs  c24   ;   alarmprint(↯fault|);
     arn p-3   , ud  p-1   ;   transfer(param);
     us  s32           LB  ;
a7:  arn b35   , ud  p-2   ; TEST:
     il        , arn b34   ;   if sense status (check) < 0 then
     hv  a6            LT  ;     go to ERROR;
e15: ncn 0     , hv (r)    ;   if only sense then return;
     arn(p-1)  , ac  p-3   ; INIT TRANSFER: param:= param + incr;
a8:  arn p-4   , ud  p     ;   transfer to or from core (param 1);
     arn p-3   , ud  p-1   ;   transfer (param);
     us  s             LB  ;
     arn(p-2)  , ar  b9    ;   length:= R:= length + 400;
     gr (p-2)  , pa  p1    ;
     hv  e13           LT  ;   if length < 0 then go to SWITCH STATUS;
     pa  b2    t 400 e10   ;   iword:= buf base + 400; go to wrapup;
     tk  30    , hv  1b2>  ;
```

```
[other input driver]
<d41,
e6:  hv  e13               ; endaction: SWITCH STATE;
[p1] paf b2    t e10  -1   ; next action: output;
     pa  a9    Vt  -399    ;   iword:= buf base - 1;
a10: gr  b4                ;   for i:= 1 step 1 until 400 do
a9:  bt  -399  t   1       ;     begin
     hv  e13               ;     iword:= iword + 1;
     hs  c71               ;
     hv  a44          NZ   ;     buf[iword] := get word;
     qq               IRC  ;     comment including marks;
     gm (b2)   t   1   MRC ;     if R ≠ 0 then alarmprint(⟨fault|);
     arn b4    , ar  b11   ;     inlength:= inlength + 1;
     hv  a10          LT   ;     if 0 ≤ inlength then begin iword:= iword + 1;
     hv (b2)   DVt  1      ;     go to wrapup end end;
                           ;   go to SWITCH STATE;

[other output driver]

d e7= i-1
[p1] pa  b2    t e10 -1    ; next action: INPUT;
     pa  a11   t   -399    ;   iword:= buf base - 1;
a11: bt  -399  t   1       ;   for i:= 1 step 1 until 400 do
     hv  e13               ;     begin
     pm (b2)   t   1   IRC ;     outword(buf[iword]);
     hs  a12               ;     outlength:= outlength + 1;
     arn b5    , ar  b11   ;     if 0 ≤ inlength then
     gr  b5                ;       go to END OTHER;
     hv  a11          LT   ;     end;
     hv  a27               ;   go to SWITCH STATE;

a12: gm  a15          MRC  ; outword: loc:= R;
     arn b11   , ar  a16   ;   bufad:= bufad + 1; us(loc, bufad + 3100);
a14: us        , gr  a16   ;
     sr  b26               ;   if blocklength - 1 < bufad then return;
     hv  s1           LT   ;   rep:= 0;
     pa  a13               ; next block:
a13: bt  0     t   -120    ;   rep:= rep + 1; if 3 < rep then
a44: qq  b41   , hs  c24   ;     alarmprint(⟨fault|);
b29: arn b28   , us  7     ;   write block(unit,paramword);
b30: arn b35   , il  23    ;   if sense(check) < 0 then
     il        , arn b34   ;     go to next block;
     hv  a13          LT   ;
     arn b27   , ac  b28   ;   paramword:= paramword + incr;
     arn a17   , hh  a14   ;   bufad:= -1; return;

a15: qq                    ;
a16: qq a15.9+1.19+3099.39 ;
a17: qq a15.9+1.19+3099.39 ;
b26: qq a15.9+1.19+3611.39 ; block length + core[a17]
b27: qq 1.9                ; increment
b28: qq 1.19+3100.39       ; paramword
>
```

```
[endactions for buffermedia]

<d41,
b34:     qq                ; status word
b35:     qq b34.9+1.19     ; get status word
b36:     qqf15.5+15.11+15.17+15.23+15.29+15.35+3.39,
[1b36]   qq b36.9+1.19     ;
[2b36]   qq 1.19           ;

a23: vk  960   , hs  e12   ; TERMINATE TAPE: SENSE
     arn 1b36  , us        ;   transfer filemark to buffer;
     pm  2b36  X           ;   param:= write filemark;
     gr  p-3   , ud  p-1   ;   transfer(param);
     us  s     , hs  e12   ;   SENSE;
     vk  d11   , ud  p-1   ;   rewind;
     us  s64   , lk  c     ;   fetch search from parametertrack;
     vk  d11   , vk (c5)   ;   fetch catalog track;
     vk (c4)   , lk  d14   ;
     vk (c4)   , hs  a20   ;   backup;
[-1] hs  a20               ;
     hv  r-1           NA  ;   while NA do backup;
     hs  a20               ;   backup;
     hs  c15           NA  ;   if NA then get word;
     pm  b14   , tl  23    ;
     arn(c15)  , ck  8     ;   store[iword]:= store[iword] ∧
     tl  16    , ck  16    ;     8m16016m  ∨ (block count pos 23);
     gr (c15)  , hs  c8    ;   sum track; write track;
     sk  d14   , hv  b46   ;   in:= out kind; go to ADJUST SPECIAL;

a20: arn(c15)  Dt  -1  IZA ; procedure backup; begin comment JJ;
     nc  d14-1 , hv  a21   ;   iword:= iword - 1; if iword < d14 then
     pm (c4)   DXt -1  M   ;     begin track:= track - 1;
     ca  -1    , ac  c5    ;     if track = -1 then begin track:= 959;
     pa  c4    t   959 NB  ;     group:= group - 1; end;
     qq (c8)   t   -2      ;     reltr:= reltr - 2;
     hs  c5                ;     read track;
     pa  c15   t   38d14   ;     iword:= d14 + 38 end;
a21: pm (c15)  , hr  s1    ;   set marks(store[iword]) end backup;

a27: arn a16   , sr  a17   ; TERMINATE OTHER:
     psn a27-1 V       NZ  ;   while bufad ≠ -1 do outword(0);
     arn b28   , hh  a22   ;   go to EXIT;
     hv  a12   X       IRC ;
>
<e18,
a28: hs  e12               ; TERMINATE DISC: SENSE;
a22: arn b24   , ck  10    ;   go to EXIT;
     tl  -28   , ar  b11   ;
     gr  b1    , hv  e14   ;
>
```

```
<d41,
e30:
[Table format:
     qq init input.9+init output.19+input driver.29+output driver.39
          + block length code.41                                          ]

[0] qq e24.9+e20.19+e0.29+e1.39  ; DRUM
[1] qq e25.9+e21.19+e4.29+e5.39, ; DISC
[2] qqfe26.9+e22.19+e6.29+e7.39  ; CARR
[3] qqfe27.9+e23.19+e2.29+e3.39, ; TAPE
>

d i=i+39, d=k-d1           ; d = no of tracks
b k=d42, i=0, a10          ;
d a1=d19-960               ; a1= group no for image
d a=d, <d35, a=1 >         ; a = no of blocks

<d39,                      ; if aux only then
d i=d2                     ;
     hs  1                 ;   begin
     hv  a6                ;
     tmove;                ;   move, b load place, move <
     qq  50,               ;
     qqf d.23+d1.39+a1.29-a1.33      ;
     tmove;                ;
     qqf,                  ;

a6:  hs  1                 ;
     hv  a7                ;
     tsetsum;              ;   setsum, move<
     tmove;                ;
     qqf,                  ;
d d2=i                     ; end else
a7:  hsf 2                 ;
×                          ;
d i=d48                    ;
     qq d35.2+11.7+d36.5+d.23+d1.39, ; load to primitive catalog;
     qq,                   ;
     tmove;                ;
d d48=i                    ;
     qqf                   ;
>                          ;
d d1=d+d1                  ;
e                          ;
e                          ; end of outermost block
[STOP, SUM]aimove
s
```

```
<-d55+1, i version
s
>

[STOP, CLEAR]

b k=d1, i=40d14, b67
d b33=d14


    b a39
[40d14] pp  -1    , vk  960   ;   procedure get param track;
        vk  d11   , lk  0     ;   begin read param track to cell 0;
        vk  0     , arn p2    ;     if first param=exit then goto HELP;
        sy  58    , sy  64    ;     LC; CR;
        hv  -9              LC ;
        pa  b16   , hr  s2    ;   end get param track;
b53:                          ; ENTRY PRINT:
<d35-2, us (-31)      t-96  > ;   rewind PRINT
b0:     qq  0     , hs  40d14 ; next print list: get param track
b60:    sy  59    , sy  0     ; entry from PAIR:
    a0: pp  p1    , arn p1    ; get first param:
        hv  -9            LC  ;   if exit then goto HELP
        hv  a8            LB  ;   if number then alarm(¢<param¢);
        hv  a5            LA  ;   if single then goto single;
        pa  a1       t e6     ;   text:=¢<undef¢;
        pmn c69   DX      IZA ;
        gp  40d14 , hs  c2    ;   store p; search;
        pa  a1        te7 NT  ;   if R≤0 then text:=¢<catalog¢;
        hs  c24           NZ  ;   if R≠0 then alarm(¢<text¢);
    a1: qq  0     , arn p1    ;
        tl  -6    , ca  -1    ;
        pp  p1    , hh  a1    ;   count p to end of name
        it  1                 ;
        qq (40d14), hs  c23   ;   writetext(name);
    a2: arn 2c    , gr  b40   ; set: set actual area;
        tl  -7    , ga  b8    ;   set kind;
        ca  0     , hv  a15   ;   goto kind 0;
<d41,   ca  1     , hv  a16   ;   goto kind 1;
        ca  2     , hv  a19   ;   goto kind 2;
        ca  3     , hv  a17   ;   goto kind 3;
>       qq  e8    , hs  c24   ;   alarm(¢<kind¢);
    a5: nc  50    , hv  a4    ; single: if -,b then goto test image;
        pp  p1    , arn p1    ;
        hh  a2            LB  ;   if base then goto set;
        arn a39   , gr  b40   ;   act.area:=buffer descrip
        pa  b8                ;   kind:=0;
    a4: arn b40   , sr  a39   ; test image and buffer:
        hv  a6            NZ  ;
        qq  a32   , hs  c23   ;   writetext(¢<buffer¢);
        arn a38   , sr  e     ;
        gr  b44   , it  1     ;   blocklength:=4096;
    a6: pa  b22   , pp  p-1   ;   boobuf:=act.area=bufferword;
```

```
        arn b39   , sr  b40   ;
        it  1             LZ  ;   booimage:=act.area=image area;
        pa  b0    , arn b0    ;
        ar  b22   , ga  b3    ;   boobuf ∨imag:=boobuf ∨booimage;
    a7: pp  p1    , arn p1    ; next param:
        ca  0     , hv  a8    ;   if 0 then alarm(‡<param›);
        ca  36    , hv  a28   ;   if m then goto set group trim;
        ca  35    , hv  a25   ;   if l then goto char per line;
        ca  22    , hv  a24   ;   if w then goto words per line;
        pa  b11   , ca  32    ;   extra:=false;
        pa  b4    , hv  a7    ;   if - then addr print:=false;
        ca  39                ;
        psn b27           IZC ;   if p begin
                              ;    begin ZA:=ZB:=true; form:=program end;
        ca  41    , psn b13   ;   if r then form:=real;
        ca  54    , psn b12   ;   if f then form:=fractional;
        ca  57    , psn b10   ;   if i then form:=integer;
        ca  55    , psn b14   ;   if g then form:=group;
        ca  19    , psn b16   ;   if t then form:=text;
        ca  51    , hv  b17   ;   if c then goto PRINT CONTROL REGISTERS
        gs  b9    V       LZ  ;   set form;
    a8: qq  e5    , hs  c24   ;   if wrong param then alarm(‡<param›);
    a9: pp  p1    , arn p1    ; after single:
        hv  a10           LC  ;   if endlabel then goto empty interval;
        hv  a11           LB  ;   if number then goto number;
        hv  a10           NA  ;
        ca  49    , hv  b18   ;   if a then goto PRINT ARITH REGISTERS
        ca  37    , hv  a29   ;   if n then goto no relative;
        ca  23    , hv  a30   ;   if x then goto extra precission;
   a10:grn b45    , grn b46   ; empty interval:
        grn b47   , pp p-1    ;   a:=bi:=bj:=0; p:=p-1;
        can(b3)   , hv  r4    ;   if boobuf ∨image then
        bs (b0)   , pm  a37   ;    R:=if booimage then 1023
        bs (b22)  , pm  a38   ;       else 4095;
    a3: gm  b48   , hv  a13   ;   else R:=blocks×blocklength-1;
        pmn b40   , tl  7     ;
        tln -23   , arn e     ;   length:=R;
        ml  b44   , hv  a3    ;
   a11:arn p1     , gr  b49   ; number:
        bs (b22)  , hv  a14   ;   if boobuf then goto set buffer interval;
        ca  0     , hv  a12   ;   if a=0 then goto no change;
```

```
        ck  20    , nc  0     ;
        hv  a12               ;   if c≠0 then goto no change;
        ck  -10   , mb  a37   ;
        ck  10    , ac  b49   ;    c:=a;
    a12:arn b49               ; no change:
        ck  -30   , mb  a37   ;
        xr        , mln b44   ;
        gm  b45   , arn b49   ;   a:=(a×blocklength+b):blocklength;
        ck  -20   , mb  a37   ;
        gr  b47   , ar  b45   ;   bj:=(a×blocklength+b)mod blocklength;
        xr        , dln b44   ;   bi:=b;
        gr  b45   , gm  b46   ;
        arn b49   , ck  -10   ;
        mb  a37   , sr  b45   ;
        gi  r3    X           ;
        mln b44   X       IZA ;
        ar  512   D       NZA ;
        pi  0     X           ;
        arn b49   , gm  b49   ;
        mb  a37   , sr  b46   ;   length:=(c-a)×blocklength+d-bj;
        ar  b49   , gr  b48   ;
    a13:arn b45   , ar  b40   ; end test interval:
        gr  b50               ;   mod area:=actual area+a;
        arn p2    , gp 40d14  ;   store p;
        hv  b2            LC  ;   if -,bmarked then goto INTERVAL
        hv  b2            NB  ;
        qq (40d14)   t 1      ;   else bi:=param;
        gr  b47   , hv b2     ;   goto INTERVAL;
    a14:tk  -20   , gr  b47   ; set buffer interval:
        gr  b46   , ar  b40   ;   bi:=bj:=b;
        gr  b50   , arn b49   ;   mod area:=area+b;
        tk  20    , ck  -20   ;
        sr  b46   , gr  b48   ;   length:=d-b;
        grn b45   , hv  2a13  ;   a:=0;
    a15:pmn b40   , tl  23    ; kind 0:
        tln -23   , dln c11   ;
        gm  b49   , ar  c11   ;
        qq  b43   , hs  b30   ;   print group number;
        arn b49   , hs  b36   ;   print track number;
```

```
        pm  a34   , gm  b44   ;   blocklength:=40;
        hv  a0                ;   goto get first param;
<d41,   124/4/12b             ;
    a16:pa  a21       ta16-2  ; kind 1: set layout;
        pmn d53   DX          ;
        cl  10    , hh  a20   ;   blocklength:=disc block length;
    a17:pa  a21     t a18     ; kind 3: set layout;
    a18:pm  a35   , hh  a20   ;   blocklength:=400;
        124/4/5/7b            ;
    a19:pm  a36   , it  a23   ; kind 2: set layout;
    a20:pa  a21   , gm  b44   ;   blocklength:=512;
        it (b14)  , pa  a22   ;   store group trim;
    a21:it   0    , pa  b14   ;   layout group trim;
b1:     arn b40   , hs  b14   ;   print various
    a22:it   0    , pa  b14   ;   reset group trim;
    a23:sy  59    , hv  a0    ;
        124/4/2/6b            ;
        4b                    ;
×b1:>                         ;
    a24:pa  a27   t    b7     ; words per line:
        pt  b5    , hv a26    ;   boochar:=false; addr:=words per line;
    a25:pt  a27   t    b6     ; char per line:
        pt  b5    t 1         ;   boochar:=true; addr:=char per line;
    a26:arn p2    , tk  30    ;
        hv  a8              NB ;   if -,number then alarm(‡<param›);
    a27:ga  0                 ;   addr:=number;
        pp  p1    , hv  a7    ;   goto next param;
    a28:it  p1    , pa  b14   ; set group trim:
        pp  p1    , arn p1    ;   set addr first param-1
        hv  r-1           LB  ;   skip b-marked
        pp  p-1   , hv  a7    ;   goto next param;
    a29:arn b27   D       IZB ; no relative: ZB:=false;
        nc  s     , hs  a8    ;   if form≠program then alarm(‡<param›);
        hv  a9                ;   goto after single;
    a30:arn s     D           ;
        ca  b12   , hv  r3    ; extra precission:
        ca  b13   , hv  r2    ;   if form≠r∧form≠f∧form≠i then
        nc  b10   , hv  a8    ;    alarm(‡<param›);
        ga  b11   , hv  a9    ;   extra:=true; goto after single;
    a32:tbuffer;              ;
    a34:40                    ;
    a35:400                   ;
```

```
    a36:512                  ;
    a37:1023                 ;
    a38:4095                 ;
    a39:qq b33.9 + 1.19      ; buffer descrip
    e                        ;
                             ;
    b a14                    ; INTERVAL:
b2:     arn b46   , tk  20   ;   set addr first cell
        gt  b26   , pmn b50  ;
        can(b8)   , hs  b21  ;   if kind=0 then get track;
<d41,   can(b8)   , hv  a2   ;
b62:    pp  c-1   , vk 960   ;   set core base;
        vk  c63   , lk  0    ;
        vk 1c63   , lk  40   ;   read init. med. to core;
        vk  0     ,grn c81-c28;  clear label check;
        grn b49   , pa  3    ;
b61:    xr  40    , pmc80-1c28;  set buffer base;
        gmc80-1c28, hs  4    ;   init. medium;
        pm c29-c28, pp (b62) ;
        hsn b37   , sr  b46  ;
        hv  a2             LZ;   skip first bj words
        hs  -26              ;
        srn e     , ar  b49  ;
        gr  b49   , hh  r-4  ;
>   a2: pa  b23     t1       ;   first:=true;
b63:   [hv  b65              ;   PAIR first run instr ]
       [hv  b66              ;   PAIR second run instr ]
    a3: pa  a9    , pa  a11  ;   words left:=char left:=0;
    a6: pp  14    , arn b9   ;   each:=14;
        ca  b16   , pp  0    ;   if t then each:=0;
        bs (b11)  , pp  21   ;   if x then each:=21;
        ca  b27   , pp  26   ;   if pn then each:=26;
        gp  a14   , ca  b27  ;
        pa  a14     t 38 LZB ;   if p then each:=38;
                             ; new block:
b3:     bs  0     , hv  a7   ;   if boobuf ∨image then goto same block;
        pa  a9    , pa  a11  ;   words left:=char left:=0;
        sy  64    , arn b45  ;   CR;
b4:     bs  1     , hs  b36  ;   if addr print then write(↟dddd↠,a);
        srn e     , ac  b45  ;   a:=a+1;
b5: a7: arn a11   , bs  0    ; same block:
        arn a9    , sr  a14  ;   if boochar∧char left>each
        hv  a10           LT ;     ∨-,boo char per line∧words left≠0
        ca  0     , hv  a10  ;      then
        nt (b28)  , pa  r1   ;
```

```
        qq  0      , hs b20    ;       begin space(sp); goto same line end;
    a9: qq  0      , hv  a12   ;
b6: a10:it  0      , pa  a9    ;   char left:=char per line;
b7:     it  1      , pa  a11   ;   words left:=words per line;
    a11:qq  0      , sy  64    ;   CR;
        can (b4)   , hv  a12   ;   if -,addr print then goto same line;
        arn b47    , hs  b36   ;   write(‹dddd›,bi);
        bs (b3)    , hv  a12   ;   if boobuf ∨image then goto same line;
        arn b46    , hs  b36   ;   write(‹dddd›,bj);
    a12:nt (a14)   , pa  b28   ; same line: printed:=-each;
b8:     can 0      , hs  b22   ;   if kind=0 then get word kind 0
        bs  (b8)   , hs  -26   ;    else get word;
b9:     hs  0      X           ;   write(form, word);
        can (b23)  , hv  a14   ;   if first∧g then
        arn b9                 ;    begin
        nc  b14    , hv  a14   ;      each:=printed+16; (i.e.printed+2);
        it (b28)   , pa  a14   ;      sp:=2;
        qq (a14)       t16     ;
        pa  b28        t-2     ;    end;
    a14:nt  0      , qq (a9)   ;   char left:=char left-each;
        qq (a11)       t-1     ;   words left:=words left-1;
        srn e      , ac  b46   ;   bj:=bj+1; bi:=bi+1;
        ac  b47    , sc  b48   ;   length:=length-1;
        pa  b23    , arn b48   ;   first:=false;
        hv  b0           LT    ;   if length<0 then goto next print list;
        arn b46    , sr  b44   ;   if bj=blocklength then bj:00;
        hv  a7             NZ  ;   if bj≠0 then goto same block
        grn b46    , hv  b3    ;    else goto new block;
    e                          ;
                               ;
    b a26                      ; FORM, NUMBER and TEXT
b10:    bs (b11)   , it  a5    ; integer:
        pa  a3         t a4    ;   set layout;
        gr  b49          IRC   ;   save marks
        gs  a26    , ann b49   ;   save s
        tl  -39    , dl  a6    ;   R:=abs(number):1000000;
        hv  a1             LZ  ;   if R=0 then goto small number;
b11:    bs  0      , hv  a     ;
        qq (a3)        t 1     ;   if -,extra then layoutadr:=layoutadr+1;
        hv  a1                 ;   goto small number;
    a:  mt  b49    , gm  b49   ;   number:=abs(number)mod1000000;
        sy  0                NT;   if pos then space;
        qq (b28)       t1    NT;
```

```
        qq (b28)       t1     ;   count printed;
        qq  a5    , hs  b30   ;   write(‹ddd ddd›,R);
        pa  b15      t 25     ;   set leading zeros in print routine;
        sy  0     , hv  a2    ;   goto second part;
    a1: can(b11)  , hv  a2    ; small number:
        qq  8     , hs  b20   ;   if extra then space(8);
        it  8     , qq (b28)  ;   printed:=printed+8;
    a2: arn b49               ; second part:
    a3: qq  0     , hs  b30   ;   write(layout,number);
        pa  b15      t 26     ;   reset print routine
        hv  a26               ;   goto write mark;
    a4: qq 6.3+6.7+1.9+1.14+6.23;              ‹-dddddd›
        qq 4.3+4.7+1.9+1.14+1.17+4.23;         ‹-dddd10d›
    a5: qq 6.3+6.7+1.14+3.23+3.27;          ‹ddd ddd›
    a6: 1000000               ;
                              ;
b12:    bs (b11)  , it  a9    ; fractional:
        pa  a7       t a8     ;   set layout;
        gs  a26         IRC   ;   store s, save marks
    a7: qq  0     , hs  b31   ;   write(layout,number);
        hv a26                ;
    a8: qq  4.3+1.7+1.9+3.13+1.14+3.17+1.19+5.23;  ‹-d.ddd10-ddd›
    a9: qq 10.3+1.7+1.9+9.13+1.14+3.17+1.19+5.23+3.27+3.31;
                              ;   ‹-d.ddd ddd ddd10-ddd›
b13:    bs (b11)   , it  a9   ; real:
        pa  a10      t a8     ;   set layout;
        gr  b49         IRC   ;   save marks; RF:=R;
        arnfb49   , gs  a26   ;   save s;
    a10:qq  0     , hs  b32   ;   write(layout,number);
        hv  a26               ;
                              ;
b14:    it a21    , pa  a15   ; group: set mask addr
        gs  a26          IRC  ;   store s;
        gr  b49   , hvn a15   ;   store R; goto take mask;
    a13:pa  a17   , arn a21   ;   skip:=0;
    a14:tk  10    , ck  0     ; next pos: mask:=mask shift 10;
    a15:arn 0          t1  LZ ; take mask: R:=mask;
        ga  a16   V       LB  ;   set group length
        pa  a17   , hv  a26   ;   exit
    a16:bs  0        t 99     ;   if grouplength>99 then begin
        ga  a17   , hv  a14   ;   skip:=grouplength; goto next pos end;
        ca  0     , hv  a14   ;   if grouplength=0 then goto next pos
        gr  a21   X           ;    else  store mask
```

```
        mkn a18   , mb  a20   ;
        tk  2     , ar  a19   ;   construct layout
        gr  b52   , pmn b49   ;
        bs (a17)  X           ;   if skip>0 then
    a17:tk  0       t -100    ;    skip:=skip-100;
        cl (a16)  , gr  b49   ;   number:=numbershift(grouplength+skip)
        hs  b30   X           ;
        qq  b52   , hv  a13   ;   write(layout,numberpart);
    a18:qq 154                ;
    a19:qq 15.3+2.7+1.14+15.23;
    a20:qq  15                ;
    a21:0b                    ;
        10/10/10/10b          ;
                              ;
b16:    bs  0     , sy  60    ; text: if lastcase=UC then UC
        hv  a24           LZ  ; again: if R=0 then goto out
        cl  34    , ck  -4    ;    digit to RA
        nc  10    , ca  15    ;    if RA=10 vRA=15
    a24:sy  58    , hr  s1    ; out:  then begin LC; go back end;
        ca  63    , ar  a25   ;    if RA=63 then RA:=64;
        ca  60    , ga  b16   ;    if RA=60 then lastcase:=UC
        ca  58    , pa  b16   ;    if RA=58 then lastcase:=-,UC;
    a25:ga  r1    , xrn       ;    writechar(RA); swap;
        sy  0     , hv  1b16  ;    goto again;
                              ;
    a26:ps  0     , arn r     ; writemark:
        ca  b19   , hr  s1    ;   if REGISTER PRINT^M then go back;
        can s-b1  , hr  s1    ;   if blockhead then go back;
        sy  27    V     NRC   ;
        sy  49    V     NRB   ;   write a b c or comma
        sy  51    V     LRC   ;
        sy  50          LRB   ;
        hr  s1                ;   go back
    e                         ;
                              ;
    b a8                      ; PRINT CONTROL REGISTERS:
b17:    pmn b39   , hs  b21   ;   get first image track
        sy  53    , arn 9b33  ;   writetext(¢<e¢);
        sy  56            LB  ;   if b marked then writetext(¢<h¢);
        qq  a6    , hs  b28   ;   write(¢p¢,cell 9);
        sy  0     , sy  0     ;   space(2);
        pmn b38   , hs  b21   ;   get last image track
        sy  39    , arn 36b33 ;   writetext(¢<p¢);
        gp 40d14  , hs  a5    ;   store p; out(p);
        sy  18    , arn 31b33 ;   writetext(¢<s¢);
```

```
        hs  a5                ;   out(s);
        sy  55    , arn 25b33 ;   writetext(‡<g‡);
        hs  a5                ;   out(group);
        sy  19    , sy  34    ;   writetext(‡<tk‡);
        arn 26b33 , tk  10    ;
        hs  a5                ;   out(tk);
        arn 26b33 , ga  a4    ;
        sy  50    , sy  24    ;   writetext(‡<by‡);
        hs  a1                ;   out and bit(by);
        sy  0     , sy  0     ;
        sy  57    , sy  37    ;   writetext(‡<  in‡);
        arn 27b33 , ga  a4    ;
        hs  a1                ;   out and bit(in);
        hv  b0            NKC ;   if -,ka∧-,kb then goto next print list
        sy  34                ;   writetext(‡<k‡);
        sy  49    V       NKB ;   if -,kb then writetext(‡<a‡);
        sy  51    V       LKC ;   if ka∧kb then writetext(‡<c‡);
        sy  50            LKB ;   if -,ka∧kb then writetext(‡<b‡);
        hv  b0                ;   goto next print list;
    a1: hs  a5                ; procedure out and bit(no); integer no;
        pa  a3        t9      ; begin out(no);
    a2: arn a4    , tk  1     ;
        sy  1     V       LT  ;   if bit then writetext(‡<1‡)
        sy  16                ;    else writetext(‡<0‡);
    a3: bt  0         t-1     ;
    a4: qq  0     , hh  a2    ; end out and bit;
        sy  0     , hr  s1    ;
    a5: hs  b28               ; procedure out(no);
        qq  a6    , sy  0     ; begin write(‡p‡,no); space(2);
        sy  0     , hr  s1    ; end;
    a6: qq 4.3+1.7+1.9+1.14+5.23; ‡p‡
                              ;
b18:    gp 40d14              ; PRINT ARITHMETIC REGISTERS:
        pmn b38   , hs  b21   ;   get last image track;
        sy  60    , sy  41    ;   writetext(‡<R‡);
        arn 35b33 , pm 32b33  ;
        sy  38            LB  ;   if overflow then writetext(‡<O‡);
        tl  39    , pm 30b33  ;
        sy  14            LO  ;   if LO then writetext(‡<*‡);
        sy  2             LO  ;
        sy  58    , hs (b9)   ;   write(layout,R);
        qq  a8    , hs  c23   ;   writetext(‡<  M‡);
```

```
b19:    arn 30b33 , hs (b9)   ;   write(layout,M);
        arn b9                ;
        ca  b13   , hv  r2    ;
        nc  1b13  , hv  b0    ;   if -,r then goto next print list
        qq  a7    , hs  c23   ;   writetext(¢<  RF¢);
        arn 32b33 , pm 30b33  ;
        grf b39   , ps  b0-1  ;
        arn b39   , hv  b13   ;
    a7: t  RF;                ;
    a8: t  M;                 ;
    e                         ;
                              ;
b20:    it (s)    , pa  r1    ; procedure space(i); integer i;
        bt  0       t-1       ;    for i:=i step -1 until 1 do
        sy  0     , hv  r-1   ;      writespace(1);
        hr  s1                ;

b21:    tl  23    , tln -23   ; procedure get track;
        dln c11   , ar  c11   ;
        tk  30    , ga  b24   ;
        cl  -10   , ga  b25   ;
        vk (b24)  , vk (b25)  ;
        lk  b33   , vk  0     ;
        hr  s1                ;
                              ;
    b a2                      ; procedure get kind 0 word;
b22:    bs  0     , hv  a2    ; begin if boobuf then goto bufferword;
b23:    bs  0     , hv  a     ;   if first then goto set;
        arn b46   , pt  b26   ;
        hv  a1            NZ  ;   if bj≠0 then goto same track;
    a:  pa  a1      t b33     ; set: set start addr
b24:    vk  0                 ;   select group
b25:    vk  0     , lk b33    ;   read track to core
        is (b25)    t1        ;   tk:=tk+1
        bs s-449    t510      ;   if tk=960
        pa  b25   , it 1      ;    then begin tk:=0; group:=group+1 end;
b26:    vk (b24)  , it  0     ;
    a1: pmn 0         t1      ; same track: get kind 0 word
        hr  s1                ;   go back
    a2: arn b50   , il  0     ; bufferword:
        srn e     , ac  b50   ;   get buffer word
        pmn b33   , hv  r-3   ;   go back
    e                         ; end;
d d19=d19-960                 ;
b38:    qq d19.29-d19.33+d16.39+25.39,
b39:    qq 26.23+d19.29-d19.33+d16.39,
b40:    qq 26.23+d19.29-d19.33+d16.39,
```

```
d d19=960d19                    ;
b43:    qq 4.3+4.7+1.14+4.23    ;
b44:    40                      ;
                                ;
                                ;
    b a21                       ;
b64:                            ; ENTRY PAIR
<d35-2, pm  r2    , ud  b53     ;   rewind;
    ah: gm  b63   , hs 40d14    ;   set PAIR instr; get param track;
×       pm  r2    , gm  b63     ;   set PAIR instr in PRINT
    a:  hs 40d14                ; next pair list: get param track
>       hv  b65                 ;   PAIR instr
        pa  b63       t b65     ;   set return to PAIR instr in PRINT
<d41,   pt  b61      tc80-1c28  ;   bufferbase:=1543;
        pa  b62       tc-1      ;   corebase:=c-1;
>       sy  1     , hs  b60     ;   writetext(¢<1¢); goto PRINT
b65:    pm  b50   , gm 40b17    ; return after first run:mod area1:=mod area
        pm c29-c28, gm 48b17    ;   set addr for init med
        pm  b45   , gm 41b17    ;   a1:=a
        pm  b47   , gm 44b17    ;   bi1:=bi
        arn b46   , gr 42b17    ;   bj1:=bj
        tk  20    , gt  a15     ;   start addr get kind 0 word 1
        arn b8    , ga  a3      ;   kind1:=kind
<d41,   ca  2     , it  a19     ;   bufferbase:=if kind=2 then 3086
        pt  b61      t a20      ;                 else 2190;
        pa  b62      t 53b17    ;   corebase:=53b17;
>       it (b22)  , pa  a11     ;   boobuf1:=boobuf;
        arn b24   , ga  a13     ;   group1:=group
        arn b25   , ga  a14     ;   track1:=track
        pm  b44   , gm 43b17    ;   blocklength1:=blocklength
        pa  a12      t1         ;   first1:=true;
        pa  b63       t b66     ;   set PAIR instr in PRINT
        hs 40d14                ;   get param track;
    a1: qq 7.2+7.27             ;   mask;
        sy  2     , hs  b60     ;   writetext(¢<2¢); goto PRINT;
b66:    pm c29-c28, gm  49b17   ; return after second run:
<d41,   it  (b8)  , bs  2       ;   set addr for init med
        hv  r5                  ;   if kind<2 then goto loop;
        arn b50   , pm  a1      ;   if kind=kind1∧unit=unit1 then
        cm  40b17 , hv  r3      ;    alarm(unit);
        qq  r1    , hs  c24     ;
        tunit;                  ;
        qq (40b17)    tb17-b33  ;    modify from buffer addr.
>   a2: pm  48b17 , pp  c-1     ; loop: set bufferbase and corebase
    a3: can  0    , hs  a11     ;   get kind 0 word 1
        bs  (a3)                ;
        hs  a18   , hs  c71     ;   initialize get word, get word;
```

```
        pa  a12           IRC ;   first1:=false;
        gm  b49           MRC ;   store word 1
        pm 49b17  , pp 53b17  ;   set bufferbase and corebase;
        can(b8)   , hs  b22   ;   if kind=0 then get word kind 0
        bs (b8)               ;
        hs  a18   , hs  c71   ;   initialize get word, get word;
    a4: pa  b23           IRC ;   first:=false;
        gm  47b17  X      MRC ;   store word
        sr  b49           IPC ;
        hv  a8            NZ  ;   if word≠word1 then out pairs
        qq (a5)     t 1   LRB ;
        qq (a5)     t 2   LRA ;
        qq (a5)     t-1   LPB ;
        qq (a5)     t-2   LPA ;
    a5: ncn 0     , hs  a8    ;   if mark≠mark1 then out pairs
    a6: arn b48   , ar  e     ;   length:=length-1
        hv  a             LT  ;   if length<0 then goto next pair list
        gr  b48   , srn e     ;   bj:=bj+1; bj1:=bj1+1;
        ac  b46   , ac 42b17  ;
        ac  b47   , ac 44b17  ;   bi:=bi+1; bi1:=bi1+1;
        arn b46   , sr  b44   ;
        hv  a7            NZ  ;   if bj=blocklength then
        grn b46   , srn e     ;    begin bj:=0; a:=a+1 end;
        ac  b45               ;
    a7: arn 42b17 , sr 43b17  ;   if bj1=blocklength1 then
        hv  a2            NZ  ;    begin bj1:=0; a1:=a1+1; end;
        grn 42b17 , srn e     ;
        ac  41b17 , hv  a2    ;   goto loop;
    a8: arn b45   , sr  45b17 ; procedure out pairs;
        ar  41b17 , sr 46b17  ;  if a≠lastprinted a
        hv  a9          LZ    ;    ∨a1≠lastprinted a1 then
        sy  64    , arn 41b17 ;    begin CR;
        gr 46b17  , hs  b36   ;     last printed a1:=a1; write(¢dddd¢,a1);
        qq  41    , hs  b20   ;     space(41);
        arn b45   , gr 45b17  ;     last printed a:=a;
        hs  b36               ;     write(¢dddd¢,a);
    a9: sy  64                ;    end;
        arn 44b17 , hs  a10   ;   CR; write(¢dddd¢,bi1);
```

```
        arn 42b17 , hs  a10   ;   write(‹dddd›,bj1);
        arn b9    , nc b27    ;
        pa  b28   V   t-34    ;
        pa  b28       t-35    ;   sp:=if p then -35 else -34;
        pan b67   X   t44b17  ;   set bi addr in program print
        arn b49   , hs  (b9)  ;   write(word);
        nt  (b28) , pa  r1    ;   sp:=-(sp+printed);
        qq  0     , hs  b20   ;   space(sp);
        arn b47   , hs  a10   ;   write(‹dddd›,bi);
        arn b46   , hs  a10   ;   write(‹dddd›,bj);
        pan b67   X   tb47    ;   set addr bi in program print
        arn 47b17 , hs (b9)   ;   write(word);
        pa  a5    , hv  a6    ;   go back
b36:a10:hs  b30               ; end;
        qq  b43   , sy  59    ;
        sy  0     , hr  s1    ;
                                ; procedure get kind 0 word 1;
    a11:bs  0     , hv  a17   ; begin if boobuf1 then goto bufferword;
    a12:bs  0     , hv  r3    ;   if first1 then goto set;
        arn 42b17 , pt  a15   ;
        hv  a16           NZ  ;   if bj≠0 then goto same block;
        pa  a16      t b17    ; set: set start addr
    a13:vk  0                 ;   select group1
    a14:vk  0     , lk  b17   ;   read track to core
        is (a14)      t1      ;   tk1:=tk1+1;
        bs s-449     t510     ;   if tk1=960 then
        pa  a14   , it  1     ;    begin tk1:=0; group1:=group1+1 end;
    a15:vk (a13)  , it  0     ;
    a16:pmn 0         t 1     ; same track: get kind 0 word 1
        hr  s1                ;   go back
    a17:arn 40b17 , il  0     ; bufferword:
        srn e     , ac 40b17  ;   get bufferword
        pmn b17   , hv  r-3   ;   go back
                                ;end;
b37:a18:gp  c68   , gp  c73   ;   initialize get word;
        it  p-1   , pa  c66   ;
        it  p-2   , pa  c67   ;
        gm  c65   , hh   s    ;
    a19:3086                  ;
    a20:2190                  ;
    e                         ;

    b a31                     ; PROGRAM PRINT
b27:    bs (b29)              ;
        sy (b29)     t 510    ;   LC;
```

```
      gr  b49          MOC  ;   store word, set leftword and abs addr
   a: gs  a22          IRC  ;   store s, halfword and f marks
      pp (b28)     t 6      ;   p:=printed:=printed+6;
      pp  p6           LOB  ;   if abs addr then p:=p+6;
      it  p5    , pa  a9    ;   char per halfword
   a1: qq  27    , pp  a29   ;   p:=table start
      tk  20    , ck  2     ;
      ga  a3    , ck  -18   ;
   a2: gt  a2    , pm  p0    ;
   a3: pin 0     X  t-253    ;   PA:=n; PB:=(); QA:=r; QB:=s; QC:=p;
   a4: tk  20          LTA  ;
      tk  10           LTB  ;
      cl  5     , tk  -15   ;
   a5: gt  a5    , ps  0     ;   s:=letter no
      bs  s-18  , ps  s-49  ; write instruction:
      bs  s-9   , ps  s-25  ;
      sy  s48   , cln -20   ;
      hv  a5           NZ   ;
      sy  54           LRB  ;   writetext(¢<f¢);
      sy  37    V      LPA  ;   writetext(¢<n¢);
   a6: sy  0     , qq  0     ;   if -,n then  space(1);
      sy  0            NRB  ;   if -,f then space(1);
      sy  0     V      NPB  ;   if -,() then space(1) else
      arn a7    , hs  a11   ;   writesymbol((();
      pm  b49   , arn a4    ;   RA:=20;
      hv  a8    X      NQC  ;
      ar  -2    D      LQB  ;   if s vp then RA:=RA-2;
      ar  21    D      LQA  ;   if r vp then RA:=RA+21;
   a7: qq  520   , hs  a11   ;   writetext( r , s or p);
      qq (b28)       t1     ;   printed:=printed+1;
      hs  b29          NQB  ;   if r then write(¢ddd¢,addrmod1024);
      qq  a28   V      NQB  ;   if r then skip next;
   a8: hs  a30             ;   write (¢ddd¢,addrpart);
      sy  0     V      NPB  ;   if -,() then space(1) else
      arn a20   , hs  a11   ;    writesymbol());
      hv  a10          LQB  ;   if s vp then goto no abs addr;
```

```
b67:    arn b47   , tk  30    ;
        ga  r2                ;
        arn b49          IQB  ;   QB:=abs addr
        ar  0      DVX   LQC  ;   if -,r v-,abs addr then goto no abs addr;
    a9: qq  0     , hv  a10   ;
        arn a23   , hs  a11   ;   writesymbol([);
        hs  a30               ;   write(¢ddd},addrpart+bi);
        arn a25   , hs  a11   ;   writesymbol(]);
        qq (b28)    t2   IPB  ;   printed:=printed+2;
    a10:pm  a17   , arn b49   ; no abs addr:
        hv  a21          NA   ;   if right halfword then goto out;
        cm  a6          NRA   ;
        hs  a24   , pm  a1    ;
        tk  10    , ps  a     ;
        gr  b49   X      MOB  ;   store right halfword;
        hh  a14   X      NRA  ;   if fullword then goto full;
    a11:ga  a13   , bs (b29)  ; procedure write symbol;
        mt  a20   , it  510   ; begin
    a12:sy (b29)    t-510 LT  ;   write case
    a13:sy  0       X         ;   write symbol
    a14:hr  s1    , tk  21    ; end;
        pp (b28)    t7    M   ; full: p:=printed:=printed+7
        gp  a9    , hs  a15   ;   write X
        qq  23    , hs  a15   ;   write V
        qq  21    , ps  a16   ;
    a15:ud  a12          NPB  ;
        pp (b28)  V t-1   NT   ; procedure write X V D;
        sy (s1)          IPB   ; begin
    a16:tk  1     , hh  s1    ; end;
        qq  52    , pp  p-4   ;   p:=p-4;
        hh  a19          LZ   ;   if indicator part≠0 then
        hs  a24   , tl  -39   ;    begin
        sy (b29)    t-510 NPB ;     space(missing space);
```

```
        tk  30    , ga  a18   ;     case
        tln 3     , ud 1a15   ;
    a17:0/-1/0/-1             ;
        tk  20    , gt  a18   ;     write indicator state
        tl  32    , ga  a19   ;
        ps  a26   , is  a27   ;
    a18:ud  s0    , ud  s0    ;    end;
    a19:ud  s0    , arn b49   ;
    a20:qqf 521   , gp  b28   ;   printed:=p;
        ca  0     , hv  a22   ;   if word=0 then goto out;
        pm  19    DX          ;   space(missing space);
        hs  a24   , hs  a11   ;   writesymbol(t);
        hs  a30               ;   write(‹ddd›,increment);
    a21:qq (b28)      t1      ; out: space;
    a22:ps  0     , hr  s1    ;   restore s ; go back;
    a23:qq  518   , it  1     ; procedure missing spaces;
    a24:it (b28)  , bs (a9)   ;
        sy  0     , hh  a23   ;
    a25:qqf 519   , hh  s     ;
    a26:pp  p-1   , pp  p-1   ; indicator letters:
        sy  50    , sy  34    ;   b, k
        sy  49    , sy  25    ;   a, z
        sy  51    , sy  38    ;   c, o
        sy  37    , sy  19    ;   n, t
        sy  35    , sy  39    ;   l, p
    a27:sy  57    , sy  40    ;   i, q
        sy  36    , sy  41    ;   m, r
    a28:qq 4.3+4.23           ;   ‹ddd›
                              ; instruction letters: table start;
    a29:qq 17.4+17.9+17.14+26.19+18.24+ 1.29+18.34+19.39
        qq 14.4+ 1.9+14.14+19.19+ 3.24+ 1.29+ 3.34+19.39
        qq  2.4+13.9+ 2.14+ 1.19+20.24+13.29+11.34+13.39
        qq 12.4+13.9+11.14+ 4.19+12.24+ 4.29+11.34+14.39
        qq 12.4+14.9+18.14+ 8.19+12.24+20.29+11.34+ 3.39
        qq 12.4+ 3.9+18.14+ 7.19+ 1.24+ 7.29+20.34+ 7.39
        qq 11.4+20.9+ 1.14+ 3.19+13.24+ 7.29+13.34+16.39
        qq 18.4+24.9+ 9.14+ 7.19+19.24+16.29+16.34+16.39
        qq  1.4+16.9+20.14+16.19+11.24+ 8.29+ 9.34+16.39
        qq 19.4+ 9.9+20.14+ 9.19+13.24+ 3.29+20.34+ 2.39
        qq 19.4+14.9+20.14+14.19+16.24+ 7.29+ 3.34+14.39
        qq 12.4+ 9.9+19.14+21.19+ 7.24+ 7.29+ 3.34+ 7.39
        qq  3.4+16.9+19.14+ 2.19+19.24+ 8.29+25.34+22.39
        qq 11.4+12.9+11.14+19.19+11.24+ 7.29+11.34+22.39
        qq 22.4+ 8.9+10.14+26.19+25.24+19.29+25.34+12.39
        qq  8.4+ 8.9+19.14+ 7.19+12.24+26.29+ 4.34+21.39
    a30:sr  983   D           ;
        it  b29           NT  ;   if number>983 then
        pa  a31       t b28   ;    write(‹ddd›,numbermod1024)
        ar  983   D           ;   else write(‹ddd›,number);
    a31:hs  0                 ;
        qq  a28   , hr  s1    ;
    e                         ;

d b28=i, b29=1b28, b30=1b29, b31=1b30, b32=1b31, b15=45b28
d b45=123i, b46=1b45, b47=1b46, b48=1b47, b49=1b48, b50=1b49, b52=1b50
```

[NUMBER PRINT ROUTINE page 1]

```
b a51                          ;
                               ;
a42: qq  0   , ck   0          ; entry address part 0≤x<1024
a50: qq  570 , tl  -30         ; entry address part -512≤x≤511
     pm  28  DV                ; entry integer
     pm  -11 D                 ; entry fractional
     pa  ra11 X    t485        ; entry real; numberpart:=true;
     gs  ra12 , gp  ra13       ;   save p; save s;
     ga  ra14 , gm ra23        ;   store exp2; store numberpart;
     pm  (s)  , arn  s         ;
     pm  (s1)              NA  ;   get layout address
a14: psn  0   X                ;   s:=exp2;
     ps  s11  , cl  -20        ;   s:=exp2+11;
     tk  14   , ga  ra17       ; unpack layout: b
     tk  10   , pa  ra16       ;   exp10:=0;
a40: it  0    , pa  ra4        ; expprinting:
     pp  256  , ck  -6         ;   bE:=256;
     ga  ra18 , tk  10         ;   h
     ck  -8   , ga  ra19       ;   f1
     tk  10   , ck  -6         ;
     ga  ra20 , tk  11         ;   d
     tk  -20  , gt  ra21       ;   -n
     tk  20   , ck  -1         ;
     ga  ra22 , tk  -6         ;   bE+f2
     ca  1    , pp  10         ;   if bE=1 then minexp:=10;
     ca  2    , pp  100        ;   if bE=2 then minexp:=100;
     tln 34   , ar  ra43       ;   group picture
     gr  ra8  , snn ra23       ;   R:=-abs(numberpart);
     pa  ra4  V  t-15   LZ     ;   if numberpart=0 then H:=-15
a49: nk  ra46 XV               ; reconversion:  else x:=numberpart×2↑exp2;
     gr  ra26 , hv  ra31       ;   goto if numberpart=0 then L3
     bs s-11  , hv  ra47       ;    else if s>0 then conversion1
                               ;    else conversion2;
a34: hv  ra48 , sr  ra26       ; round x2: R:=rounded x;
a17: pa   0   XVD  t11  NO     ;  if -,overflow then b:=11
     mt  ra7  , hv  ra49       ;    else goto reconversion;
     bs (ra4) , ntn (ra4)      ;   if H>0 then begin R:=0;
a21: qq (ra18)                 ;    h:=h-H end else h:=h-n;
a19: pp   0   , gt  ra13       ;   p:=f1; a13incr:=R;
     bs  p509 , hv  ra36       ;   if p≠3 then goto count h
a11: arn 485  D   t-485        ;   exppart:=-,numberpart; numberpart:=false;
     hs  ra5               LT  ;   if exppart then write ₁₀
     bs (ra24), arn ra23       ;   if b1>0∧x<0 then
```

```
     arn  -480 DV       NT ;   R:=-
     arn  32   DV          ;    else R:=+
     bs  p510  , ck  10    ;   if R<0∧p<2 then R=small
     hs  ra37  , pp  3     ;   write sign
a36:                       ;
a18: bt  0           t-1   ; count h:
     hsn ra2   , hv  ra36  ;   write space before digits
     bs  p509  , hv  ra11  ;
a13: pp   0    , ncn  0    ;   restore p;
     hsn ra1   , qq        ;
     bt (ra4)      t-1     ;   count H
     hsn  ra   , hv  r-1   ;   write digits before point
     arn ra38  , bs (ra20) ;   if d>0 then
     hs  ra5   , it  -1    ;    write point;
a20: bt  0     , hh ra39   ;   count d, write decimals
a12: ps  0     , xrn       ;   restore s; M:=0;
     bs (ra24) , pm ra16   ;   if b2>0 then M:=exp10;
     can (ra22), hr  s1    ; EXIT
     gm  ra23  , srn  ra9  ;
     pm (ra22) DX          ;
     ps  9     , hh  ra40  ;   goto expprinting;
a37: ca  p     , hh  s     ;   if p=0∧Raddr=0 then go back
     gr  ra26  , arn ra9   ;   sign not counted in group
     ac  ra8   , arn  ra26 ;
a5:  bs  (ra)  , hv  ra1   ;   if b1>0 then write digit
     mb  ra44  , ga  ra6   ;
a39: hvn ra10  , arn ra3   ;   else if digit≠0 then write digit
a4:  it  0          t1     ;   count H                          else write 0
     bs  0     , hvn  ra5  ;   if H<0 then write 0
a:   bt  0         t-1     ;   count b1
     mln ra3   , tk  30    ;   next digit in R
a1:  ar  16    D        LZ ;   zero instead of space
a2:  ga  ra6   , bs (ra50) ;   if actual case=upper then
     mt  ra7   , it  510   ;     R:=-R;
     sy (ra50)   t-510  LT ;   write case;
a10: qq (ra42)      t1     ;   count printed;
a51: arn ra8   , sr  ra9   ;   actual group:= actual group-1;
     gr  ra8   , nc  -273  ;   if -,group full then write out
a6:  sy  0     , hhn  s    ;    else if out>58 then begin
     tk  4     , it  58    ;    actual group:= next group;
     bs (ra6)  , hv r1a51  ;   write out end
```

```
a44: sy  -256  , ud  ra10  ;    else begin writespace; count printed;
                           ;    actual group:=next group-1; write out end;
a41: hh  ra51  , it  1     ; conversion:
a28: qq (ra4)       t-1    ; begin comment      (count H)
a25: ps  s-3   , nk  r1    ;   by multiplication by 2↑3/10 or 10/2↑4
     ps  s0    , gr  ra26  ;   x is converted to form
     pm  ra26  , bs (ra46) ;   x=x2×10↑H where x>x2≥.1
a47: mkn ra27  , hh  ra41  ; conversion 1:
a46: tk  s     , gr  ra26  ;
a48: ps  s7    , ar  ra29  ; conversion 2:
     pm  -1    DV       LT ;
     mkn  ra30 , hv  ra28  ; end conversion;
     ps  (ra4) , can (ra22);   s:=H; if bE=0∧f2=0 then begin
     bs  s-15  , it  64    ;   if s>15 then bE=1 end;
a22: ca  0     , hv  ra31  ;   if bE∧f2=0 then goto L3;
     arn  ra17 , sr  ra20  ;   R:=b-d;
     ga  ra32  , sr  ra18  ;   a32:=b-d; Raddr:=b-d-h-1;
a7:  mb  -1    DX       LT ;   M:=if b-d-h-1<0 then b-d-h-1 else -1;
a38: xr  315   , it (ra4)  ; L1: R:=M;
a32: bs  0     , hs  ra33  ;   if b-d>H then change exp10
     mt  ra7   , it (ra18) ; L2: R:=-R;
     bs (ra4)  , hs  ra33  ;   if H>b then change exp10
a31: arn ra20  , ga  ra    ; L3: b1:=d;
a45: ar  ra4   , ga  ra24  ; L4: b2:=H+d;
a24: bs  0     , ga  ra    ;   if b2>0 then b1:=b2;
     it (ra17) , bs  (ra24);   if b<H-d then
     arn ra17  , hh  ra45  ;    b2:=b; goto L4;
     arn  256  D        NT ;   if b2≥0 then R:=.5;
a35: ps  (ra24), pm  ra21  ; rounding: s:=b2;
     bs  s511  , hh  ra34  ;   if s≤0 then goto round x2;
a16: xr  p0    , mln ra29  ;   R:=R×.1;
     ps  s-1   , hh  ra35  ;   s:=s-1; goto rounding;
                           ; change exp10:
a33: ac  ra16  , bs (ra16) ;   exp10:=exp10+R; if exp10≥minexp then
     sc  ra4   , hh  s-1   ;    begin H:=H-R; goto L1 ∨L2 end;
     sc  ra16  , hv  ra31  ;   else exp10:=exp10-R; goto L3;
                           ;
a3:  10                    ;
a9:  qq   1                ;
a27: can  s409 , cm (r-410);   0.8
a29: vy   p51  , mln (204) ;   0.1
a30: qq   320              ;   10/16
a43: qq  -17.5+1.25-1.39   ;
d a8=i, a26= 1i, a23=2i    ;
e                          ;
```

```
i=39i, d=k-d1                ;
b k=d42, i=0, a10            ;
a1=d19-960                   ;   a1=image group
a2=40d14                     ;   a2=first cell in core
a3=b53, a7=b64               ;   a3=entry print, a7=entry pair
a=d, <d35, a=2>              ;   a=number of blocks
<d39,                        ;   if aux only then
i=d2                         ;    begin
    hs 1                     ;
    hv a4                    ;
<d36,                        ;     if aux reserved then
    tres;                    ;      res, no of blocks
    qqf a.39                 ;
x<d39,                       ;     else
    tset;                    ;      set,
    qqf d35.39               ;        aux kind,
    qqf a.39                 ;        no of blocks,
[STOP, SUM]ai base print, pair
s                            ;        typein
[STOP, CLEAR]                ;
><d39,                       ;
    qq 39,                   ;   concat pid 0,
    qq 57,                   ;
    qq 52,                   ;
    qqf                      ;
    tprint;                  ;   print, spec<
    qqf d.9+a3.19+a2.29      ;
    tpair;                   ;   pair, spec<
    qqf d.9+a7.19+a2.29      ;
    qqf,                     ;
a4: hs 1                     ;
    hv a5                    ;
    tmove;                   ;   move, b loadplace, print<
    qq 50,                   ;
    qqf d.23+d1.39+a1.29-a1.33;
    tprint;                  ;
    qqf,                     ;
a5: hs 1                     ;
    hv a6                    ;
    tsetsum;                 ;
    tprint;                  ;
    qqf,                     ;
d2=i                         ;
a6: hsf 2                    ;   end
xi=d48                       ;   else load to primitive catalog
    qq d35.2+11.7+d36.5+d.23+d1.39,;
    qq,                      ;
    tprint;                  ;
    qq d.9+a3.19+a2.29       ;
    tpair;                   ;
    qq d.9+a7.19+a2.29       ;
d48=i                        ;
    qqf                      ;
> d1=d+d1                    ;
e                            ;   end image load
e                            ;   end print, pair
[STOP, SUM]ai print, pair
s                            ;
```

```
[Call:   run, <name> <

<name> ::= <name of drum area>|<empty>

      This aux program will execute a translated ALGOL program
      described by <name>. If no parameter is present the ALGOL
      program will be taken from the work area]


[Here follows STOPCODE,CLEARCODE]

b k = d1, i = 238, a10, b6 ;
d a=15                     ;
b1:  pa  8     t   1       ; ENTRY:  set no init of cells[0:9];
a1:  pp  d13-1 , hs  c52   ;   p := start parameterlist - 1;
[1]  qq  e5    , hs  c24   ;   if parameter ≠ name then alarm(‡<param‡);
a4h: hh  a3    , mb  a5    ;   if parameter = end list then goto NO PARAM;
     nc  0     , hh  1a1   ; NAME:
     tl  -16   , gr  261a  ;   if kind ≠ 0 then alarm(‡<param‡);
     tln 16    , gr  264a  ;   tracks occupied in top of free:= bits(8,23,areaw);
     ar  a8    , gr  263a  ;   first track program := bits(24,39,areaword);
     arn 264a  , ar  261a  ;   execution end trac := first track program + 1;
     sr  a8    , sr  b3    ;   if first track program ≤ last track image
     qq        V       LT  ;     ∧ last track program ≥ first track image
     arn  b4   , sr  264a  ;   then set half inhibit
     grn -1    V       LT  ;   else set no inhibit;
     acn -1            MA  ;
     vy  d43   , arn 2c    ;   select help alarm output unit;
     tl  -4    , nc  7     ; TEST BITS: if reserved bit of areaword = 0
     nc  5     , grn 261a  ;                ∨ special bit of areaword = 0 then
     pm  264a  , hsn c3    ;   tracks occupied in top of free := 0;
     lk  265a  , vk (c4)   ;   to core(first track program,265e4); wait drum;
     arn a6    , sr  267a  ;   if instruction[2] of program
     pa  1a1   t   a7      ;      ≠ lk  p  ,  hh  r-2
     hh  1a1           NZ  ;   then alarm(‡<not present‡);
     hh  1a1           NA  ;
     hh  1a1           LB  ;   goto instruction[0] of program;
a3h: hv  265a  , hsn c2    ; NO PARAM:    get free;
     lk  d14   , vk (c4)   ;   to core(first catalog track,d14); wait drum;
     arn 3d14  , gr  2c    ;   R := areaword := work; goto NAME;
     hh  a4                ;
                           ;
a5:  qq  -1.39-31.7        ;   mask;
a6:  lk  p     , hh  r-2   ;
a7:  tnot present;         ;
a8:  qq  1.39              ;   1
d b6=d19-960, b5=25d16     ;   group no for image; last track image;
b3:   qq d16.39+b6.29-b6.33;   first track image;
b4:   qq  b5.39+b6.29-b6.33;   last track image;
      qq [fill]            ;
d b=259a                   ;
b:                         ;
[259e4] qq [lower limit in buffer.39]            ; EXECUTION PARAMETERS:
[260e4] qq [initial last used in buffer] 4096.39 ;
[261e4] qq [tracks occupied in top of free.39]   ;
[262e4] qq [abs track look up] c64-1             ;
[263e4] [execution end track.39]                 ; not loaded;
[264e4] [first track program.39]                 ; not loaded;
<j,i load error, run
s
>
```

```
i=i+39, d=k-d1                  ;
b k=d42,i=0,a10                 ;
a1=d19-960                      ; a1=group no for image
a=d,<d35,a=1>                   ; a=no of blocks

<d39                            ; if aux only then
i=d2,hs1                        ; begin
     hv a5                      ;  (if aux reserved then
<d36,tres;                      ;
     qqf a.39                   ;   res,no of blocks,
x<d39                           ;    else
     tset;                      ;    set,aux kind,no of blocks,
     qqf d35.39                 ;     type in)
     qqf a.39                   ;
[after i follows STOP,SUM and a sum character]
iabase,run
s
[STOP,CLEAR]
><d39                           ;
   qq  39,                      ; concat pid 0,
   qq  57,                      ;
   qq  52,                      ;
   qqf                          ;
   trun;                        ; run,spec<
   qqf d.9+b1.19+b1.29          ;
   qqf,                         ;

a5: hs 1                        ;
    hv a6                       ;
    tmove;                      ; move,b loadplace,run<
    qq 50,                      ;
    qqf d.23+d1.39+a1.29-a1.33  ;
    trun;                       ;
    qqf,                        ;

a6: hs 1                        ;
    hv a7                       ;
    tsetsum;                    ;
    trun;                       ;
    qqf,                        ;
d2=i                            ; end else
a7: hsf 2                       ;
x                               ;
i=d48                           ;
   qq d35.2+11.7+d36.5+d.23+d1.39,; load to primitive catalog;
   qq,                           ;
   trun;                         ;
   qqd.9+b1.19+b1.29             ;
d48=i,qqf                        ;
>                                ;
d1=d+d1   ;
e         ; end image load
e         ; end run
[after i follows STOP,SUM and a sumcharacter]
ia run
s
```

[STOP, CLEAR]
<-d55+1, i version
s
>

[Navne globale til SLIP

	før SLIP	efter SLIP
c	plads for search i HJÆLP	
c2	indhop i search	
c4	vk kanal i search, indhop til pak op og vælg	
c5	vk gruppe i search	
c7	lk-ordre til læs katalog	
c18	bruges ved indhop i search	
c30		første kanal i SLIP
c31	. .	antal kanaler i SLIP
c32		c2-c
c33	. .	c4-c
c34		c5-c
c35	. .	c7-c
c36		2c18
c37	. .	2c
c38		indhop i SLIP
c39	. .	antal kanaler som hentes med HELP
c63	init medium kanal	
c70	baandfejl	
c71	get word from buffer	
c77	return if sum	
d1	første frie kanal	første fri kanal efter SLIP
d2	første fri celle for init	samme efter slip
d9	første kanal i HJÆLP	
d10	ferritlagerbillede-320	
d11	parameterkanal	
d12	plads i HJÆLP til kanal med adressetrykprogram	
d14	kanalplads for inputstreng i HJÆLP	
d15	længde af texttrykprogram	
d17	by-værdi for l	
d19	den gruppe SLIP læser til	
d20	plads for SLIP i FL	
d21	første katalogkanal	
d22	antal kanaler/320	
d24	i adressetrykprogram	
d25	-	
d26	-	
d27	-	
d28	-	
d29	-	
d30	-	
d31	-	
d32	work in free	
d35	medium kind	
d36	reserved	
d37	texttryk kanal-d9 d37-1 er search kanal	
d39	input mode	
d40	adressetrykprogram i HJÆLP	
d47	return to init help	
d48	init katalog	
d50	image gruppe under indlæsning af slip]	

m
<d35, c30=d1, × c30=6d1> ; gør plads til 6 kanaler hvis tromle
c32=c2-c, c33=c4-c, c34=c5-c, c35=c7-c, c36=2c18, c37=2c

b k=c30, i=d20, a24, b21, c18, d7, e123
e64=d9, e88=d10, e103=d17, e98=d19
d=-27, d1=d-2, d3=d1-7, e38=d3-40, e84=e38-d15, e123=40d12-d40
e95=e84-e123, e39=e95-40, e50=e39-41

b a [navne i adressetrykprogram]
a=d40-e95, e81=d24-a, e80=d25-a, e77=d26-a, e79=d27-a, e78=d28-a, e32=d29-a
e33=d30-a, e83=d31-a
e

[navne paa variable]
a14=3d, e66=a14, a15=1a14, e47=1a15, e45=1e47, d5=1e45, a17=8d5, e30=a17
e48=1a17, a23=1e48

d6: qq 63.25
qq 1.1+3.16+7.24+5.32
qq 1.26+1023.39

d7: e8=d7-1 [OP-code table]

; Differens tabel	1.sym 2.sym værdi	1.sym 2.sym værdi	
qq 512.9		-55.9-34.15	
qq 512.9 +55.9+34.15	+57.27+35.33+44.39	-57.9-19.15+37.21	; il, it
qq -57.27-35.33-44.39+57.9+19.15-37.21	+57.27+18.33+36.39	-56.9-56.15+60.21	; is, hh
qq -57.27-18.33-36.39+56.9+56.15-60.21	+56.27+41.33+17.39	-56.9-34.15+34.21	; hr, hk
qq -56.27-41.33-17.39+56.9+34.15-34.21	+56.27+21.33+56.39	-56.9-18.15+50.21	; hv, hs
qq -56.27-21.33-56.39+56.9+18.15-50.21	+55.27+57.33+29.39	-55.9-55.15+46.21	; gi, gg
qq -55.27-57.33-29.39+55.9+55.15-46.21	+55.27+51.33+47.39	-55.9-49.15+22.21	; gc, ga
qq -55.27-51.33-47.39+55.9+49.15-22.21	+55.27+41.33+21.39	-55.9-39.15+42.21	; gr, gp
qq -55.27-41.33-21.39+55.9+39.15-42.21	+55.27+36.33+26.39	-55.9-34.15+54.21	; gm, gk
qq 55.9+34.15		-39.9-57.15	
qq 512.9 +39.9+57.15	+55.27+19.33+23.39	-55.9-18.15+61.21	; gt, gs
qq -55.27-19.33-23.39+55.9+18.15-61.21	+52.27+35.33+14.39	-52.9-34.15+13.21	; dl, dk
qq -52.27-35.33-14.39+52.9+34.15-13.21	+51.27+49.33+25.39	-51.9-36.15+38.21	; ca, cm
qq -51.27-49.33-25.39+51.9+36.15-38.21	+51.27+35.33+20.39	-51.9-34.15+19.21	; cl, ck
qq -51.27-35.33-20.39+51.9+34.15-19.21	+50.27+19.33+39.39	-50.9-18.15+49.21	; bt, bs
qq -50.27-19.33-39.39+50.9+18.15-49.21	+49.27+51.33+ 6.39	-49.9-50.15+ 9.21	; ac, ab
qq -49.27-51.33- 6.39+49.9+50.15- 9.21	+49.27+41.33+ 2.39	-49.9-37.15+ 4.21	; ar, an
qq -49.27-41.33- 2.39+49.9+37.15- 4.21	+40.27+40.33+ 0.39	-39.9-57.15+35.21	; qq, pi
qq 39.9+57.15		-35.9-34.15	
qq 512.9 +35.9+34.15	+39.27+51.33+48.39	-39.9-49.15+32.21	; pc, pa
qq -39.27-51.33-48.39+39.9+49.15-32.21	+39.27+39.33+31.39	-39.9-36.15+27.21	; pp, pm
qq -39.27-39.33-31.39+39.9+36.15-27.21	+39.27+19.33+33.39	-39.9-18.15+30.21	; pt, ps
qq -39.27-19.33-33.39+39.9+18.15-30.21	+37.27+51.33+43.39	-37.9-35.15+16.21	; nc, nl
qq -37.27-51.33-43.39+37.9+35.15-16.21	+37.27+34.33+15.39	-37.9-19.15+41.21	; nk, nt
qq -37.27-34.33-15.39+37.9+19.15-41.21	+37.27+18.33+40.39	-36.9-50.15+ 8.21	; ns, mb
qq -37.27-18.33-40.39+36.9+50.15- 8.21	+36.27+35.33+12.39	-36.9-34.15+11.21	; ml, mk
qq -36.27-35.33-12.39+36.9+34.15-11.21	+36.27+19.33+10.39	-35.9-34.15+52.21	; mt, lk
qq 35.9+34.15			
qq 512.9	+35.27+24.33+59.39	-25.9-40.15+ 1.21	; ly, zq
qq -35.27-24.33-59.39+25.9+40.15- 1.21	+25.27+35.33+62.39	-25.9-33.15+57.21	; zl, zj
qq -25.27-35.33-62.39+25.9+33.15-57.21	+23.27+41.33+28.39	-21.9-34.15+55.21	; xr, vk
qq -23.27-41.33-28.39+21.9+34.15-55.21	+21.27+24.33+51.39	-20.9-52.15+63.21	; vy, ud
qq -21.27-24.33-51.39+20.9+52.15-63.21	+20.27+18.33+45.39	-19.9-35.15+18.21	; us, tl
qq -20.27-18.33-45.39+19.9+35.15-18.21	+19.27+34.33+24.39	-18.9-51.15+ 7.21	; tk, sc
qq -19.27-34.33-24.39+18.9+51.15- 7.21	+18.27+41.33+ 3.39	-18.9-37.15+ 5.21	; sr, sn
qq -18.27-41.33- 3.39+18.9+37.15- 5.21	+18.27+34.33+53.39	-18.9-24.15+58.21	; sk, sy
qq -18.27-34.33-53.39+18.9+24.15-58.21			

```
e19: ar        D           ; r
e62: d2: pa    D   t3      ; m
[1e62] 40                  ;
[2e62] 153                 ;
[3d2]  qq 1.3-1.15         ;
[4d2]  qq e39-1            ;
[5d2]  qq                  ;
[6d2]  qq 1.39             ;

       [ordrelæsning]
b a12, b8                  ;
e4:  gk        VD          ; nyt ord med mulighed for adr tryk.
[1e4]pa  e4                ; nyt ord uden mulighed for adr tryk.
     ps  6e4   , pi        ;
     pi (e4)   IK t-20     ;
     grn d     XM          ;
     pi        , grn e27   ; forbered ny linie
[6e4]hh  e20   ,           ; første symbol
e29: gr  1d3   , hv  p+a1  ; gem 1 op hopordbog
e7:a1:hv e28   , hv  e9    ; 0-9 tal, adr
     hv  a2    , hv  e9    ; a-eik, op ell lab, adr
     hv  e28   , hv  e9    ; +- tal, adr
     hv  e28   , hv  c1    ; . 10 tal, fejl
     hv  a3    , hv  b1    ; , /CR terminator1, 2
     hv  a4    , hv  b2    ; AXS)...op, mod
     hv  a4    , hv  b3    ; mprst(op, start adr,
     hv  a4    , hv  c1    ; =:h op fejl
a2:  it (4d3)  , pa  e5    ; pos:= y
     hs  e20               ;
     bsp +511  , hh  a11   ; test g = 0, label m index
     bs  p-6   , hvn a12   ; test g = 7 label
e93: hv  a5    , sy  64    ; op, skriv vr efter s
     pa  e83   , hv  e4    ; klar til kanalnr. udskrift

a3:  ca  64    , hv  e4    ; blank CR
b1:  ca  64    , hv  a6    ; linie slut
     hv  c1          LTB   ; test om hhao tilladt
     bs  s-2   , hv  c1    ; test om for mange adr
     pi  32    t   -33     ; sæt hhao
     hs  e20   , qq 1      ; læs 1 op
     gr1 d3    , ca  64    ; test for linie slut
a6:  ps  e4-1  , HV  e43   ; gem 1 op, helord lagres
a4:  hs  e20               ; læs resten af op
a5:  ck  -6    , ac  1d3   ; op karakterer til 1d3
     hs  e36               ; hent op-tabel
     arn 1d3   , pm  e19   ; R:= op karakterer, M:= relativmærke
     pa  a7    V t  e8     ; op-opslag
     it  9           NO    ;
a7:  ar            t  1    ;
     hv  a7-1          NT  ;
     gm  e27   , pm  3d2   ; sæt relativmærke, maske i M
     cm  5d2   , hh  a8    ; hop hvis ikke venstre tabel
```

```
a9:  ck  -4    , mb  d6    ; operation paa plads
a10: ck  -10           LPA ; evt hhao
     ab  d                 ; til helord
     gr  d             MA  ;
     hsn e20           X   ; læs 1 adrcif
a8h: hhp +a1   , ck  18    ; hopordbog
     cm1 d3    , hv  c1    ; forbudt operation
     hv  a9                ; operation OK
b2:  pm  1d6   , it (4d3)  ;
     bs  44    , hvn b4    ; , -, SnfF)
     ca  54    , itn -17   ; fF
b5:  pi  80    LZ t-65     ; Fmærke, forbyd hhao ell blind
     tln(4d3)  V           ; skifts
b4:  hv  b5            NPA ; hop til OK hvis -, hhao
     hv  c1            LZ  ; fejl hvis 0
     mb  2d6   , hv  a10   ; accumuler bit

b3:  ca  19    , hh  b6    ; t
     bs  s-1   , hv  c1    ; , fejl(tælledel)
     ca  8     , hv  b7    ; (
b8:  arn 4d3   , ca  4     ; , sæt rel mærke
     hv  c1                ; fejl hvis ( eller t
     ga  e27   , HV  e10   ; læs adr m. e20
b7:  hs  e20               ;
     pt  e27   t   4       ; sæt (
     bsp +505  t   510     ;
     hv  b8                ; g = 6
b6h:e68:hve9   , grn e27   ; læs adr, slet rel mærke
     bs  s+510 , ps  s+2   ; , sæt tælledellæsn
[navn]a11h:hve10,hs  e24   ; læs heltal, ciffer læst
a12: bs  p505  , hv  c1    ; g ≠ 7
     gr  d     , it (e5)   ;
     bs  3     , hv  e53   ; i eller k
     pa  e49   t   1       ;
     arn 4d3   , ca  47    ; ,h
     pa  e49   , hs  e20   ;
     arn 4d3   , ga  e52   ;
     ca  48    , hhr +1    ; :
     hv  e12   , arn d5    ;
     ck  30    , hs  e51   ;
     hv  1e4               ;
e
```

```
b a20    [adresselæsning]
e17: arn d3    , ca  64    ;
     hh  s                 ;
e10: hsn e20   X           ;
e9:  pa  e25   , gm  1d1   ; fortegn, adr:= 0
     gm  d1    , it  s+509 ; term s = 0
a11: pt  a1                ; def, ciffertæller
     hsn e22   X           ; læs et led
     bsp +509  , hvp +a2   ; p = 0 ∨ 1 ∨ 2
a12: bss +509  , hvn a3    ; s = 0 ∨ 1 ∨ 2 udhop ordre
     bss +507  , hv  c1    ; s = 3 ∨ 4 for mange adresser
     arn 1d1   , it  512   ;
     bsp +508  , hv  c1    ; fejl hvis terminator ≠ , ∨ / ∨ CR
e15:a18:hrs+1  , qq  1     ; bruges som konstant
a2:  pp (2d3)  , hv  a5    ; , heltal
     pp (e5)   , hv  a6    ; , navn
e25: [fortegn] , hv  c1    ; , fejl(++)
```

```
a6:  bs  p-2   , hv  a7    ; , a - e
     hv  c1            NZ  ; cifre efter i eller k
     arn p+d5  , hv  a8    ; hent i eller k

a7:  hs  e16               ; opslag
     hv  a9            NRA ; udefnavn
     tl  -30   , pp (e27)  ; , rel msk
     cap -2    , sr  d5    ; , -i
a8:  ar  d1    , pp (2d3)  ; + term,
a14: bsp +508  t   510     ;
     bs (4d3)  , hh  a10   ; , .
a15h:tk  30    , mt  e25   ; adr paa plads, fortegn
     ac  1d1   , it  510   ; adresse
a13: bsnp+510  , hv  c1    ; fejl hvis navn følger term
     gr  d1    , hv  a11   ; term = heltal ell 0
     tk  30    , mt  e25   ; term var heltal
     ac  1d1   , hv  a12   ; . udhop
a5:  bsp +509  , hvp +a13  ; for heltal: switch paa termin.
a10h:hv  a14   , gr  d1    ; , efter
     hsn e26   X           ; læs rent heltal
     tk  20    , gt  r+1   ;
     arn d1    , ns        ;
     cls +39   , hh  a15   ; skift term, akkumuler

         [udef navn]
a9:  pp (2d3)  , nt (e25)  ;
     bs  1     , it (e47)  ;
a1h: bs (e5)   , it        ;
     bs  1     , hv  c2    ; forbudt udef. navn
     bsp +508  , hv  c2    ;
     bss       , ck  10    ;
     ga  1d1   , tl  -30   ;
     arn d5    , tl  30    ; i
     bss       , ck  -10   ;
     ck  20            LRB ;
     GR (e5)   , hvn a12   ; sæt i ordbog, udhop ordre]
a3:e27:arn     D           ; relativmærke ind
     ck  20    , ab  1d1   ; , ; , adresse
     bss       , cl  -10   ; , hhao
     pss +2    , pt  d     ; tæl s, tælledel = 0
     ab  d     , gr  d     ; helord, helord
     grn e27   , pm  d3    ; relativmærke, symbol
     hhn p+e7  X           ; hopordbog b

         [opslag]
e16: tk  30    , ck  -1    ;
     ar  a18   , ga  a16   ;
     gt  a17   , it  d5    ;
e5:  arn       , ga  r     ;
     ca  512   , hv  c3    ; ikke erklæret
     gt  r     , it  0     ;
e18h:a16: bs   , hv  e5    ; hop index > længde
     gr  e30   , it (a16)  ;
a17h:arn(e5)   , bs        ;
     ck  20    V       IRA ;
     pi  5     V IQA -264  ;
     pi  8     V IQB -266  ;
     pi  2       LB  -3    ;
     ck  20    , hr  s+1   ;
```

```
         [h = def adr eller = def adr]
e13: hsn e20   X           ; læs ciffer
e11: pa  e52   , arn 4d3   ; redef. lovlig
     ca  49    , hv  e10   ; læs defineret adresse
     hv  c1                ;
e
```

```
b a20
e22: arn 4d3   , pp (2d3)  ; læs et adresseled
     bsp -2    , hr  s+1   ; p ≥ 3 udhop
a18: gp  3d3   , ga  5d3   ;
     bs  p     , hv  a16   ; p ≠ 0
     hs  e24               ; p = 0 læs heltal
     pp (3d3)  , hr  s+1   ;

a16: bs  p-1   , hv  a17   ; p = 2 fortegn
     it (4d3)  , pa  e5    ; p = 1 navn
     hs  e26               ;
     pp (3d3)  , hr  s+1   ;

a17: it (e23)  , pa  e25   ;
     hs  e20               ;
     bs  p-1   , hr  s+1   ;
     hv  a18               ;

e24: arn 4d3   , ck  -30   ;
     ml  e95+e123-2        ;
e26: hs  e20               ;
     bsp +511  , hv  e24   ;
     hrn s+1   X           ;

         [Den centrale læsesekvens]
         [læs et symbol]
e14: arn d3    , ca  64    ; som e20 men udhop for CR
e96: qq [checksum], hhs    ;
e20: pp        , lyn d3    ; comment situation
[1e20]ga d3    , tk  -1    ; lige-ulige
     ga  a2                ;
     bs (a2)   NT  t 32    ;
     hh  a3                ;
     gt  a1    , it  e1    ;
a1:a2: arn     , it        ;
     bs        , ck  10    ; skift for ulige
e23: ck  -10[case], tk -7  ; case tilføjes
     ga  2d3   , sr  2d3   ; g
     ck  7     , ga  4d3   ; y
     nc  124   , hh  a20   ; hop ikke sum code el. clear code
a20: hvn e41   , is (d3)   ;
     it  s1    , qq (e96)  ; summer e96:= e96 + char - 63
e2:  bs        , hv  a4    ; evt fejludskrift
     nc  p92   , hv  a12   ;
     pa  e32   t   511     ;
     sy  29    , can(e83)  ;
     arn 1d5   , hs  e81   ;
     arn d5    , hs  e81   ;
     ps  6e4   , sy  62    ;
a15: qqn 64    X           ; bruges i input fra tromle
a12: ca  126   , hh  e20   ; blind ogsaa i strenge
     bsp -19   , hv  a5    ; , comment eller streng
```

```
a11: pp (2d3)  , it  100   ;
     bs (4d3)  , hv  a5    ; internt symbol
a10: arn d3    , hr  s+1   ; symbol, udhop
a5:  ca  125   , hh  a6    ; caseshift
     bs  p-250 , hv  pa8   ; streng eller comment
     ca  123   , hv  a9    ;
e91: ca  122   , pp  300   ; ; skip til VR
     ca  121   , pp  301   ; [ skip til ]
     ca  p102  , ppn 302   ; × skip til × eller >
     ca  p-148 , hv  s1    ; skip til VR efter fejl
a19: a8=a19-300, e82=a10-a8
     ca  p-200 , hv  a11   ; comment slut VR eller skip til ,/:
     ca  120   , hv  c5    ; fejl
     ca  p-200 , hv  e20   ; comment slut ] eller >
a3h: hh  e20   , ck  2     ; blind, > 64 eller paritetsfejl
     ca  254   , hh  e20   ; blind hvis alle huller
a6:  hv  c5    , nt (e23)  ; , caseshift
     pa  e23   , hh  e20   ;
a9:  can s-e29+1, hv e6    ; _ indhop fra e4
     can s-d6  , hv  e6    ; _ indhop efter b
     ca  se97  , hv  a10   ;
a4h: hv  e20   , ca  100   ; _ blind, skriv copy efter i el. fejl
a13: nc  100   , it  -1    ; tæl ikke somme linier
     qq (e2)   , ga  a13   ;
     sy (d3)   , hv  a12   ;

b a2     [input fra disc eller tromle]
e117:bt (-3)   t   1       ; tæl karakterer
     gm  6d3   , hv  a     ; næste ord
     arn -4    , ck  -6    ; næste karakter
a2h: gr  -4    , tk  -4    ;
     ca  63    , arn a15   ; CR
ah:  hv  1e20  , arn -5    ;
     pa  -3    t   -4      ;
     nc  39e50 , hv  a1    ; hop hvis samme kanal
     arn 6d2   , ac  -2    ; tæl kanaler
e101:is [kanal]t   1       ;
     bs  s-449 t   510     ;
     pa  e101  , it  1     ; næste gruppe
e100:vk [gruppe], vk(e101) ;
     lk  e50   M           ; læs næste kanal
     vk (e101) , it  -39   ;
a1:  pmn(-5)   V LA  t1    ; hent næste ord hvis tromle
     hs  20e50             ; ellers hop
     hv  e121          NZ  ; hop hvis baandfejl
     cl  -6    , gm  -4    ;
     pm  6d3   , hh  a2    ;
e
e
         [i = definition]
b a3
e53: hv  c1            NZ  ; index efter i
     ncn(e5)   , hv  c1    ; k definition udenfor blokhoved
     hs  e11   , qq  e4    ; læs def. adr.
     cl  -30   , hv  e42   ; sæt il
```

```
         [navn = def. adr.]
e12: it (e5)   , pt  a2    ; gem bogstav
     pa  e52   , hs  e10   ; redef. lovlig, læs def. adr.
a2h: pa  e5                ;
     ps  e4    , hv  e51   ; hop til sporing

         [erklæring]
e85: it (4d3)  , pa  e5    ;
     bsn(e5)   X   t2      ;
     can p-1   , hv  e26   ;
     hv  c4                ;
e
         [sæt i]
e42: gr  a15               ; a15:= i ny
[+1] srn e47   , tk  -20   ; ibt
     ar  a15           IPC ; iny-ibt
     tl  -10   , tln -29   ;
     dln 1e62  , ar  e47   ; R:= ent((iny-ibt)/40)+k0
     mb  e90   , gr  a14   ;

         [ændring]
e46: vke98[gruppe],vk(e44) ; vælg gruppe, vælg kl
     sk  e39   , arn a14   ; skriv kl, R:= kny
     pa  e45               ; frisk:= false
     hv  e37           LPB ; indhop
e21: gr  1d5   , pm  a15   ; kl:= kny, M:= iny
     gm  d5    , ck  -10   ; il:= iny, Radr:= kl
     ga  e44   , srn e47   ; e44:= kl
     tk  -20   , ar  d5    ;
     tl  -10   , tln -29   ;
     dln 1e62  X           ;
     vk (e46)  , vk (e44)  ;
a10: pa  e45   LZ  t1      ;
     gr  2d5   , ck  -10   ;
     ar  4d2   , ga  e57   ;
     arn e44   , ca (b11)  ;
     sk  e38   , ud  a3    ;
a16: lk  e39   , vk  960   ;
e76: hr  s1    , pp        ; g vælg gammelt medium
     arn e111  , hh  e41   ;

         [gemning]
e43: arn d                 ;
e57: gr [celle]MPC t1      ;
     arn 6d2   , ac  d5    ; il:= il + 1
     ar  2d5   , pa  e45   ; iv:= iv + 1, -, frisk
     gr  2d5   , sr  1e62  ;
     hr  s1            LT  ; udhop hvis kanal ikke fuld
     vk (e46)  , vk (e44)  ;
     sk  e39   , arn 6d2   ;
e44: vk[kanal] t   1       ;
     ac  1d5   , hvn a10   ;
```

```
         [understregning og betingelser]
e6:  bs (2d3)  , hv  e94   ; hop hvis betingelse
     hs  e20   , qq  6e4   ; læs
     ca  50    , psn e34   ; b begin
     ca  52    , hv  1e4   ; d blind
     ca  53    , psn e35   ; e end
     ca  54    , udn e62   ; f flydende
     ca  55    , hh  e76   ; g gammelt medium
     ca  56    , hv  e58   ; h hjælp
     ca  57    , hh  e92   ; i informer operatør
     ca  34    , psn e104  ; k karakter input
     ca  36    , pan e62   ; m maskintal
     ca  37    , pan e19   ; n normal ordre input
     ca  39    , psn e89   ; p parametre
     ca  41    , udn e56   ; r relativ ordre input
     ca  19    , psn e105  ; t text input
     ca  20    , hv  e73   ; u udhops adresse
     ca  22    , psn e72   ; w sæt work
a2:  ca  s-6e4 , hv  1e4   ;
[1a2]arn s     V       LZ  ; kanal beskrivelse i R
     pm  -2    V       IPC ; eller gammelt medium i M
     ps  s-1   , hv  e     ; hent kanal og hop
     qq        V       NPC ; skip hvis gammelt medium er ydre
     gm  40e50         MPC ; ellers gem gammelt medium
     pm  e65               ;
     gm  -2    M           ;
     ca  35    , hh  a4    ; l strimmel input
     sy  58    , sy  64    ; s skrivemaskine input
     pa  -2    t   1       ;
a4:e75:pae2    , ud  -2    ;
     pm  e107  , gm  e20   ;
e92h:hv  1e93  , it  1     ; , i
     pa  e2    , sy  64    ;
     sy  58    , hhn e91   ;

e56: pa  e19   t   2       ; sæt r
e107:pp        , lyn d3    ; e20 værdi ved læsning fra ydre enhed
e90: 1023                  ;

b a2     [vælg discinput]
a2:  gr (e31)  , hv  e31   ; gem et textord i stack
e89: qq  c16   , it  a2    ; modificer udhopsadresse
     pa  e119  , it (e31)  ; og gemmeadresse i textlæseprogram
a1:  pa  a     , ud  d6    ;
e97=-i, hs e20 , ps  a1    ; læs char
     bs (e23)  , hv  r-1   ; skip chars in upper case
     nc  17    , hv  e122  ; fortsæt hvis ikke <
e111:arn c14   D           ;
e55:a: pp      , hh  e41   ; hent resten af vælg disc og hop
e
```

```
;                          Slip karakterlayout.
; y |                              g 2d3
;4d3| 0       1       2       3       4       5       6       7       |4d3
;  0| 0       .i      .-+     .10     .       . 5.0   .m      .       |  0
;  1| 1       .k      .       ..      .       .B      .s      .       |  1
;  2| 2       .j      .       .       .       .A      .r      .       |  2
;  3| 3       .a      .       .       .       .K      .p      .       |  3
;  4| 4       .b      .       .       .       .Z      .t(     .       |  4
;  5| 5       .c      .       .       .       .T      .       .       |  5
;  6| 6       .d      .       .       .       .M      .       .       |  6
;  7| 7       .e      .       .       .       .N      .       .       |  7
;  8| 8       .       .       .       .       .D      .       .       |  8
;  9| 9       .       .       .       .       .V      .       .       |  9
; 10|         .       .       .       .       .X      .       .       | 10
; 16|         .       .       .       .       .C      .       .       | 16
; 18|         .       .       .       .       .O      .       .       | 18
; 19|         .       .       .       .       .Q      .       .       | 19
; 21|         .       .       .       .       .L      .       .       | 21
; 26|         .       .       .       .       .R      .       .       | 26
; 34|         .       .       .       .       .P      .       .       | 34
; 44|         .       .       .       .       .I      .       .       | 44
; 45|         .       .       .       .       .nS     .       .       | 45
; 46|         .       .       .       .       .f)F    .       .       | 46
; 47|         .       .       .       .       .       .       .h      | 47
; 48|         .       .       .       .,/     .       .       .:      | 48
; 49|         .       .       .       .       .       .       .=      | 49
;100|         .       .       .       .cr CR  .       .       .       |100
;101| ]       .       .       .       .       .       .       .       |101
;102| ×       .>      .       .       .       .       .       .       |102
;109| ml ML   .       .       .       .       .       .       .       |109
;118| |       .       .       .       .       .       .       .       |118
;119|  119.0  .       .       .       .       .       .       .       |119
;120|  120.0  .       .       .       .       .       .       .       |120
;121| [       .       .       .       .       .       .       .       |121
;122| ;       .       .       .       .       .       .       .       |122
;123| _       .       .<      .       .       .       .       .       |123
;124| sum     .clear  .       .       .       .       .       .       |124
;125| up LOW  .       .       .       .       .       .       .       |125
;126|  126.0  .       .       .       .       .       .       .       |126
;
;119.0 blinde: stop end rød tab off on blaa
;              STOP END RØD TAB OFF ON BLAA
;120.0 forbudte: i lc: 10 15 26 42 45 46 47 65
;                i uc: 10 15 26 42 45 46 47 65
;126.0 blinde ogsaa i text: tf low TF UP
;  0.5 resten: aa uvwxyzloqøæg^ AA vUWYJØÆEGH
```

[karaktertabel]

```
e1:;            lige                    ulige            nr lige        ulige
;      lower case   upper case      lower    upper          low   up    low   up
    qq 0.32+109.39+ 0.12+109.19+ 0.2+  1.9+ 5.22+  0.29; 0 ml    ML    1     ∨
    qq 0.32+  2.39+ 0.12+102.19+ 0.2+  3.9+ 4.22+ 48.29; 2 2     ×     3     /
    qq 0.32+  4.39+ 7.12+ 49.19+ 0.2+  5.9+ 0.22+122.29; 4 4     =     5     ;
    qq 0.32+  6.39+ 0.12+121.19+ 0.2+  7.9+ 0.22+101.29; 6 6     [     7     ]
    qq 0.32+  8.39+ 6.12+  4.19+ 0.2+  9.9+ 5.22+ 46.29; 8 8     (     9     )
    qq 0.32+120.39+ 0.12+120.19+ 0.2+119.9+ 0.22+119.29;10               stop  STOP
    qq 0.32+199.39+ 0.12+199.19+ 5.2+  0.9+ 5.22+  0.29;12 end   END   aa    AA
    qq 0.32+123.39+ 0.12+118.19+ 0.2+120.9+ 0.22+120.29;14 _     |
    qq 0.32+  0.39+ 5.12+  0.19+ 2.2+123.9+ 1.22+102.29;16 0     ∧     <     >
    qq 6.32+  1.39+ 5.12+ 45.19+ 6.2+  4.9+ 5.22+  5.29;18 s     S     t     T
    qq 5.32+  0.39+ 5.12+  0.19+ 5.2+  0.9+ 5.22+  9.29;20 u     U     v     V
    qq 5.32+  0.39+ 5.12+  0.19+ 5.2+  0.9+ 5.22+ 10.29;22 w     W     x     X
    qq 5.32+  0.39+ 5.12+  0.19+ 5.2+  0.9+ 5.22+  4.29;24 y     Y     z     Z
    qq 0.32+120.39+ 0.12+120.19+ 4.2+ 48.9+ 3.22+  0.29;26               ,     ₁₀
    qq 1.32+124.39+ 1.12+124.19+ 0.2+119.9+ 0.22+119.29;28 clear CLEAR rød   RØD
    qq 0.32+119.39+ 0.12+119.19+ 0.2+119.9+ 0.22+119.29;30 tab   TAB   off   OFF
    qq 2.32+  0.39+ 2.12+  0.19+ 1.2+  2.9+ 5.22+  0.29;32 -     +     j     J
    qq 1.32+  1.39+ 5.12+  3.19+ 5.2+  0.9+ 5.22+ 21.29;34 k     K     l     L
    qq 6.32+  0.39+ 5.12+  6.19+ 5.2+ 45.9+ 5.22+  7.29;36 m     M     n     N
    qq 5.32+  0.39+ 5.12+ 18.19+ 6.2+  3.9+ 5.22+ 34.29;38 o     O     p     P
    qq 5.32+  0.39+ 5.12+ 19.19+ 6.2+  2.9+ 5.22+ 26.29;40 q     Q     r     R
    qq 0.32+120.39+ 0.12+120.19+ 5.2+  0.9+ 5.22+  0.29;42               ø     Ø
    qq 0.32+119.39+ 0.12+119.19+ 0.2+120.9+ 0.22+120.29;44 on    ON
    qq 0.32+120.39+ 0.12+120.19+ 0.2+120.9+ 0.22+120.29;46
    qq 5.32+  0.39+ 5.12+  0.19+ 1.2+  3.9+ 5.22+  2.29;48 æ     Æ     a     A
    qq 1.32+  4.39+ 5.12+  1.19+ 1.2+  5.9+ 5.22+ 16.29;50 b     B     c     C
    qq 1.32+  6.39+ 5.12+  8.19+ 1.2+  7.9+ 5.22+  0.29;52 d     D     e     E
    qq 5.32+ 46.39+ 5.12+ 46.19+ 5.2+  0.9+ 5.22+  0.29;54 f     F     g     G
    qq 7.32+ 47.39+ 5.12+  0.19+ 1.2+  0.9+ 5.22+ 44.29;56 h     H     i     I
    qq 0.32+126.39+ 0.12+125.19+ 3.2+  1.9+ 7.22+ 48.29;58 low   LOW   .     :
    qq 0.32+125.39+ 0.12+126.19+ 0.2+124.9+ 0.22+124.29;60 up    UP    sum   SUM
    qq 0.32+119.39+ 0.12+119.19+ 0.2+126.9+ 0.22+126.29;62 blaa  BLAA  tf    TF
    qq 4.32+100.39+ 4.12+100.19+ 0.2+120.9+ 0.22+120.29;64 cr    CR
```

```
        [begin]
e34: qq  c11   , hv  d6    ;
     arn a17               ;
     gr (e5)   , it (a18)  ;
b6:  pt (e5)   , hs  e31   ;
     grn(e31)  M           ;
a18:e63: arn   D   t-1     ;
     hh  b6            NZ  ;
     hs  e31               ;
     hv  b1                ;

        [sporing]
e51: ga  e48   , arn d     ; e48:= il, R:= index
     hs  e16           IOB ; hent, OB:= 0
     hv  e52           LRA ; defineret
     pa  a17   , pt  a17   ;
b2:  gr (e5)   , it (e48)  ; definer bit 1-10
     pa (e5)   , hh  1i    ;
```

```
b4:  arn a21   , ga  a20   ; næste til R
     ca        , hv  b3    ; færdig hvis 0
     ck        , tl  -30   ; id til pos 39
     ar  a17   X       IOA ; k:= ent((Ref + id)/40)
     dln 1e62  , ck  -10   ; ks til Radr
     ca (e44)  , hvn b10   ; ks = kl
b11: ca[kanal]960, hv a19  ; ks = gammel ks
     vk (e46)  , vk (b11)  ; vælg gammel ks
[2b11]it(b11)  t   512     ;
     bs  448   , sk  e38   ; skriv gammel ks hvis < 960
     ga  b11   , vk (b11)  ;
     lk  e38   , vk  960   ; læs ks
b10h:a19:arn a7, ar  a16   ;
     ga  a9    , cln -10   ;
     ac  a9    , arn(a9)   ;
     ck  -10   V       NOB ; if r ∨ p then OA:= 1
     ck  -2    V       NA  ;
     ck  -2            IOA ;
     ck  12    , ga  a21   ; næste adr gemmes
     gr (a9)   , it (e48)  ; c[id] ændres
     pa (a9)   , ck  29    ;
     pi        LO  t511    ; if p then OA:= 0
     tk  31                ;
e49: bs        , hhn b8    ; ikke h
a24: qq        V       NOB ; p, tælledel
     hhn b8            NA  ;
     ck  -4    , pm  6d2   ;
     ca  56    , tl  28    ; HV -> HH
     ca  32    , tl  26    ; PA -> PT
b8:  tl  27    , ar (a9)   ;
a20: sr        D       LOA ; -r
     ck  -10           LOB ; 10 - 19
a9:  gr        , hv  b4    ; paa plads
b3:  arn(e5)   , ck  10    ; ordbog
     pi  256   V NOB -257  ; 0 - 9 eller 10 - 19
a7:  qq e38-e39V           ;
a21: qq        , hv  b2    ;
a1:  ck  20            NRB ; ordbog paa plads
     gr (e5)   V       MQC ;
e74: nc (e48)  , hv  c6    ;
     hr  s+1               ;

         [redefinition]
e52: bs        , hv  e74   ; kontroldefinition
     gr (e5)   , it (e48)  ;
     pa (e5)   , it (e48)  ;
     pt (e5)   , arn(e5)   ;
     ck  20    , hv  a1    ;

         [tæl op i stak]
e31: it  e71   t   1       ;
e116:bs  e39-1 , hr  s1    ;
e58h:hv  c9    , pa  e110  ; stakoverløb, h udhop til help
     pi  16    , hv  e46   ;
```

```
         [udhopsadresse og end]
e73: ps  e4    , qq  d6    ; u, ved e hoppes retur til e35, d6 til s
e35:e109:qq c12, pm (2d)   ; e endkanalnr. R40, 41:= ref40, 41
     hsn e20   X       IPC ; indicer ref i P, læs char
     ca  64    , hv  s     ; hop hvis CR
     hs  e9                ; læs def adr
     ck  10                ; enhed i pos 39
     gr  e70           MB  ; sæt udhopsadr i exitparam
     hv  s             IPB ; indicer udhop i PB og hop

e61: gr  a14   , tln 10    ; resten af end
     gr  a15   , hv  e46   ;

e3:      [bufferord eller kanalvalg]
<d35,qq e38+40.19+201.39   ; 40 ord til e38
     qq d6+40.19+201.39    ; 40 ord til d6
×    ga  3e3               ;
[1e3]vk        V       NT  ; vælg gruppe
[2e3]vk  -1    , it  -64   ;
[3e3]vk        , hr  s1    ; vælg kanal
>
         [hent kanal til d6]
e36: arn c18   D           ; OP-kanal
e:   ca  c18   , hr  s1    ; udhop hvis samme kanal
<d35,ga  e     , tk  -31   ;
     ar  1e3   , il        ; læs fra buffer
     hr  s1                ;
×    hs  e3                ; vælg kanal
     ga  e     , lk  d6    ;
     vk (3e3)  , hr  s1    ;
>

         [beting]
e94: hs  e10   , qq  6e4   ;
     bs (1d1)  , hv  e4    ;
     pp  302   , hh  e20   ;

e72: qq  c10   , hv  e86   ; sæt work
e65: vy  e103  t   -8      ; e33 værdi ved læsning med l
e104:qq  c16   , hv  d6    ; karakterinput
e105:qq  c16   , hh  e54   ; textinput

e37: vk (b11)  , ud  2b11  ; afslut slip
     sk  e38               ; skriv sporingskanal hvis nødvendigt
     vk  960   , vk  d11   ;
     sk  e110  , arn -2    ;
     qq (-5)   LA td14-e50 ;
     hv  -9                ; skriv paramkanal, og hop til help.

e110:qq  2                 ; udhopsparametre til help
     texit;                ;
e70: qqf       ,           ; her sættes evt. udhopsadresse
     qqf       ,           ;
```

```
ba      [tallæsning og fejlreaktioner]
e28: arn e60   , hs  e     ;
c1h: hv  d6    , hs  a     ; 1. syntax læs til ,/: el CR og hop e4
c2:  hs  a     , hv  e4    ; 2. forb. udef. samme reaktion
c3:  hs  a                 ; 3. ikke erklæret -      -
c4:  hs  a                 ; 4. gal erkl. læs ,/: el CR og hop 1b1
c5:  hs  a                 ; 5. ubenyttet symb. fortsætter
c6:  hs  a     , qq  e4-1  ; 6. kontrol def.        -
c7:  hs  a                 ; 7. talfejl syntax      -
c8:  hs  a                 ; 8.    -    størrelse   -
c9:  hs  a     , qq  a2    ; 9. stak overløb læser fra skrivem.
e121:hs  a     , qq  a2    ;10. medium eksisterer ikke    -

a:   arn c13   D           ; fejl
         [hent kanal til e38]
e41: ar  e108  , vk (e46)  ;
     vk (b11)  , ud  2b11  ; vælg evt. sporingskanal, test om i brug
e60: qq  c15   , sk  e38   ;
a3:[1e60] pa b11 t  960    ;
e120:<d35,tk -31 , ar  e3  ; dan bufferord
     vk  960   , il        ; læs fra buffer
×    hs  e3                ; vælg kanal
     lk  e38   , vk (3e3)  ; læs kanal og vent
>e108:qq c10   , hv  e38   ;
e

b a21    [tallæsning]
a6=e95+e123-2

a17: hsn e20   X           ;
a3:  pa  a7            IZB ; sæt ikke decimaler, og førcifre
     ca  32    , it (e23)  ;
     pa  a4    , ca  32    ; sæt fortegn
a8:  hs  e20           IZB ;
     bs  p     , hr  s+1   ; udhop hvis ikke cifre
     ARS 4d3   , CK  10    ;
a7:  PP        , ML  a6    ; tal:= tal × 10 + ciffer
     PP  p1          NZ    ; korriger for flere cifre
     dl  a6    X     NZ    ; end der kan være i M
     CA  Sp    , HV  a8    ; ingen faktor korrektion
     GM  d     , PM  d1    ;
a15: MKS pa5   , NK  a14   ; .8 eller 10/16
     BS  p     , IT  4     ;
a14: IT  t-3               ;
a2:  XR        , IT  -1    ; korriger exp2
a1:  BT        , HV  a15   ; kun effektiv under 10potens omr.
     GM  d1    , PM  d     ;
     CAS s-e   , HV  a16   ; udhop efter potens opregn.
a9:  HV  a8    , GM  d     ; her starter læsning af 10potens
     HS  a17               ;
     PP (a4)   X   9       ; if pos exp then p:= 1
     BS  p     , PP  1     ; else p:= -1
     PM  d1    , SR  2d2   ;
     HS  c8          NT    ; fejl exp for stor
     AR  2d2   , CK  -10   ;
     GA  a1    , HH  a2    ; sæt exp 10; hop til potens omr.
a19: PA  a18   t 40    LZB ; Gruppe læsning
     ARS d     , TK  10    ;
     GR  d     , HS  a17   ;
     XR        , MT  a4    ;
```

```
a18: TK  t-10             ;
     NS (a18)  , CK  s    ;
     AC  d     X          ;
     CA  3     , HV  a19  ; ny/
a13: PP (2d3)  , ARS a5   ; Talslut,
     BS  p-1   , HV  a    ;
     IT (4d3)  , CK (4d3) ;
     GA  a20   , IT  -49  ;
a20: PI        , PP  4    ; sæt mrk.
a:   HS  e43              ; sliplagring
     BS  p+508 , HS  c7   ; fejl terminator
     PM  d3    X          ;
a21: CA  64    , HV  e4   ; CR, udhop
     HS  e20   , QQ  e    ; læs evt. blinde terminatorer
     HV  1d6              ;

<d35, b a9, b4 [initialicer slip]
a6:  qq 1i+2.19+1546.39   ;
a7:c38:arn c37-5, ar c37-4 ; dan bufferord
e71:a8:tk  20  , ck -20   ;
a:   ac  e3    , ac  1e3  ;
     ac  a1    , grn a    ;
a5:  arn c37-4 , ar  6d2  ;
     hh  a9           NZ  ;
     arn a6    , il       ;
     arn a8    , ac  a7   ;
     ac  c37-5 , arn a6   ;
a9h: us        , hs  c71  ; get word
     xr        V       LZ ;
     hv  c70              ; baandfejl
     ar  2     D       LA ;
     ar  1     D       LB ;
     ac  a     , it  -1   ;
     bt  e67   , hv  a5   ;
<d35-2,us(-31) t   -96>   ; rewind
<d35,arn a                ;
     hv  c77          NZ  ; sumfejl
     arn a1    , us       ; gem op tabel i buffer
× b a4, b4                ;
a:   vk        V          ;
c38: arn c37   , ar  a1   ; beregn abs kanal af sidste kanal i slip
e71: tl  -16   , tln -23  ;
     dl  b     , ar  b    ; e71 = staktop ved indhop,
     ck  -10   , ga  1e3  ; sæt gruppenr
     ga  a     X          ;
     sr  a1    , pm  a    ;
     gm  1e3   V       NT ;
     it (a)    , qq (2e3) ;
     ck  -10   , ac  e72  ;
     ac  e111  , ac  e108 ;
     ac  e34   , ac  e109 ;
     ac  e36   , ac  e89  ;
     ac  e104  , ac  e105 ;
     ac  e60   , ac  e    ;
>
```

```
     grn e71-2 X       MC  ; sæt slut mærke i første celle af stak
a3:  gm  d3-1  M   t1      ; nulstil arbejdsceller
     bt  26    t   -1      ;
     vk  960   , hv  a3    ;
     vk  d11   , lk  e38   ; læs parameterkanal
     vk  d37+d9, arn 1e38  ; hent parameter
     ar  960   D       NT  ; adder 960 til gruppe nr hvis mindre end 512
     arn a2    V       LA  ; hvis parameter ≠ helpnumber hent image
     arn a2            NB  ; beskrivelse
     gm  e71-1 , ga  e46   ; sæt gruppe nr i e46
     ga  b11   , it (b11)  ;
     pt  1e60              ;
     tk  10    , gt  e47   ; ibt.19 til e47
     gt  2d    , cl  10    ; og 2d
     gm  e66   , gt  d     ; kb.39 til e66, i.19 til d
     sr  d     , ck  10    ; ibt.39 ren til R
     gr  1e66  , mt  b4    ; ibt.39 til 1e66
     ml  1e62  , gm  e30   ; 40 kb-ibt til e30
     arn e66   , pm  b2    ;
     gm  40e50 M           ; sæt ingen gammel streng
     ac  e47   , lk  e84   ; ibt.19+(40kb-ibt).39 til e47, læs text kan.
     vk  2d9   ,lke39+e123 ; læs taltryk kanal
     hs  e21   , ps  512   ; start tromleblok, 512 til s
     gs  3d5   , gs  4d5   ; 512 til 3d5, 4d5, ..., 7d5
     gs  5d5   , gs  6d5   ;
     gs  7d5   , arn d     ;
     pa  2d    IPC te 71-2 ; 2d:= e71 - 2.9 + ibt.39
     pa  e47   IQC te 71-1 ; e47:= e71 - 1.9 + ibt.19 + kb.39
b4:  ck  -20   , hs  e42   ; sæt il
     hv  b1                ;

b2:  pa  e92   V te 121    ;
a2:  qq e98+294d10.19+10.39;

a1:  b3=40i-j, c=1k

<d35, d4=-480, e40=80, c39=1c-c30
xd4=6c-c30, e40=1
>
```

```
b k=c, i=e38, a7 [mediumvalg]
c14=d4, d4=d4+e40, e69=k

     arn -2            IPC ;
     qq        V       NPC ;
     gr  40e50         MPC ; gem gammelt medium hvis internt
     bs  p1    , hv  a     ; = gammelt medium
     it (e111) , pa  e     ; indicer d6 tom
e87: ps  r-1   , arn [e59] ; fyld sidste ord i navn op
     hv [e122]         NOA ;
     hs [e122]             ;
     vk  960   , it  e50   ;
     bs (e31)  , hv  c9    ; ikke plads i lager
     gp  e31   , pp  p-1   ; afstak navn
     vk  d37+d9-1, lk d6   ; læs searchkanal
     pmn c36   DX      IZA ;
     vk  960   , it  e50   ;
     pa  c35+d6, hs  c32+d6; search for mediumname
     arn 1.2   VD      LZ  ;
     hv  e121              ; name is not in katalog
     ar  2d6   , vk  960   ;
     hv  e121          LT  ; fejl hvis outputmedium
     arn 2d6   , vk  c63   ;
     qq  e50   , lk  d6    ; læs init medium
     vk  960   , hs  d6    ; init medium
     hv  e121          NZ  ; baandfejl
a6:  pm  -2            IQB ;
     hv  a2            NC  ; hop hvis ydre medium
     pa  e116  t 1e  50    ;
     dln ra4   , ar  ra4   ;
     ck  -10   , ga  e100  ; gem gruppenr
     vk (e100) , cl  -10   ; vælg gruppe
     qq (-5)   NQC te50-d14; kanalplads til e50
a3:  ga  e101  , vk (e101) ; gem og vælg kanal
a7:  lk  e50   , vy  16    ;
     pm  ra5   , vk  960   ;
     gm  e20   , hv  1e93  ;
a:   pm  40e50         IQC ;
     gm  -2            MQC ;
     hh  1a6               ;
a2=e75
a4:  960                   ;
a5:c=1k, pp    , hv  e117  ;

<i-40e38,:>; længde af vælg medium
b=a4-e38+b3, b1=a6-e38+b3
e
e
```

[her kan indskydes flere kanaler]

```
<d35, × d4=0, c39=c-c30, c=c30-6>

b k=c, i=d6, b7 [tallæsning]
c15=d4, d4=d4+e40

     ARS d3    , HH  b     ;
     BS  p+511 , HV  b1    ; test for blinde,
     BS  p+510 t   -511    ; terminatorer
b:   hv  a21   , PA  a1    ; set exp10
b1:  PM  256   DX      IZB ; sæt cifre ikke læst
     GR  d1    , PA  a2    ; sæt 10-potens faktor og exp2
     HSS a3    X           ; læs evt. fortegn og cifre
     BS  p509 -512     IPC ; sæt NPC, test for . og 10
     XR        , HH  b2    ; rent heltal læst
     BS (e23)  , HV  b3    ; 10 uden.
     pa  a7    t   -1      ; sæt efter .
     HS  a8                ; læs decimaler
     HS  c7            NZB ; fejl . uden cifre
b3:  PM  6d2           NZB ; 2↑(-39), 10 uden cifre
a4:  XR        , MT  a4    ;
     PM  e23   X           ;
     CA  p+7   , HH  a9    ; 10
a16: MLS d1    , NL  b7    ; heltal × 10 potens faktor
     HV  b4            LZ  ; tal = 0;
b7:  IT        , PP (a2)   ;
     BS (e62)  , HV  b5    ; flydende
     TL  1             ITA ;
     TK  -1    , GR  d     ;
     AN  nd            NTA ;
     BS  p+39  , HS  c8    ; fejl, maskintal for stort,
     TK  p+40              ;
     SR  6d2           LO  ; korriger +1,
b4:  GR  d     V           ; maskintal lagres
b5:  NKF p+40  , GRF d     ; flydende tal lagres
     HS  c8            LO  ; fejl, maskintal for stort
b6:  hs  c7            NZB ; endelig lagring
b2:  hv  a13   , MT  a4    ; fejl, uden cifre
     GR  d     X           ; heltal
     CA  3     , HV  a19   ; test for gruppe
     BS (e62)  , HV  a16   ; test for f, i = 370
     BS (a2)   , HS  c8    ; fejl, heltal for stort
     HV  b6                ; hop til test for cifre og lagring
     can s409  , cm (r-410);
a5:  qq 1.10+7.15          ;
c=1k, qq 320               ;
<i-40d6, :>; længde af læs tal
e
e
```

```
         [begin]
b k=c, i=d6, a5
c11=d4, d4=d4+e40

     hs  e20               ; læs tegn, understregning inform
     ca  64    , hv  e4    ; , ny linie (ordre)
b12: ca  34    , pi  32    ; , k
     arn 1d5               ;
     gr  a14   , tk  10    ; Kny:= Kl
     ar  2d    , ar  d5    ;
     gr (e31)          MPC ; pak[ref, ib, kl, il]
     arn d3    , ca  34    ; , k
     ps  a3    , hv  a4    ;
a1:[9d6] ca 57 , hh  a     ;
b17: arn d5    , hh  a5    ; Iny:= Il
b9:a4: can(e45), arn 6d2   ; if -, frisk then Kl:= Kl + 1
b18:a3:ac  1d5 , hv  e13   ; læs = def. adr.
     cl  -30   , gr  a14   ; kny:= læst
     hs  e14   , hv  b17   ; if CR
a:   hv  a1    , hs  e13   ;
b19h:a5h:cl -30, gr  a15   ; il:= Iny
     tk  20    , gr  2d    ;
     it (e31)  , pa  2d    ; ref:= n
     hs  e31   , qq  a2    ; tæl n
     hv  1e42          NPA ; ikke tromle
     arn e47   , gr (e31)  ; gem tromleref
     arn a14   , ar  2d    ;
     gr  e47   , it (e31)  ;
     pa  e47   , hs  e31   ; pak[reft, ibt, kb]
a2:  hs  e46           IPC ; ændring
     arn e47   , tl  -30   ;
     tln 10    , tk  19    ;
     tl  -19   , mt  r     ;
     ml  1e62  , gm  a17   ; Ref:= kb × 40 - ibt
b1:  can p-4               ; test
     hs  e14   , hv  1e93  ;
     hs  e85               ; max index i R
     pm (e5)   X   td5     ; R:= stak[i], e5a:= i
     gr (e31)  , sr  2d    ; stak[n]:= stak[i]
     hv  c4            NT  ; if start[i] > ref d.v.s. dobbelt erklæring
     ncn p-4   , hv  c4    ;
     it (e31)  , pa  a17   ;
     cln -11   , ar  e15   ;
c=1k,ga  e63   , hv  1e34  ;
<i-40d6,:>; længde af begin
e
```

b k=c, i=d6, b7 [end]
c12=d4, d4=d4+e40

```
     gk  e4    , ps  b2    ;
     vy        , arn e47   ;
b1:  vy  32    LKA t-33    ;
     vy  16    LKB t-17    ;
     sy  64    , ud  e79   ;
b2:  hh  s     , hs  e95   ;
     arn 2d    , hs  e80   ;
     arn d5    , hs  e81   ;
     arn 1d5   , hs  e81   ;
[330]pt  e18   t   b5      ;
     pp  2     , it (2d)   ;
b6:  pa  e31   , arn p1d5  ;
     ga  a24   , pp  p1    ;
     gan b     , it (a24)  ;
     bs (e31)  , hv  b4    ;
b3:  gp  e5    , hs  e16   ;
     gr  d     , vy        ;
     hv  r4            LRA ;
     tl  -20               ;
[340]vy  16            NZ  ;
     hv  b             LZ  ;
     hs  b1    , sy  p46   ;
     arn b     , hs  e33   ;
     arn d     , hs  e77   ;
     hv  b             LRA ;
     arn d     , hs  e80   ;
b:   arn       D   t1      ;
     ck  -30   , hv  b3    ;
b5:  arn(a24)  , gr  pd5   ;
b4:[350]bs p505, hh  b6    ;
     pt  e18   t   e5      ;
     hs  b1    , arn(e47)  ;
     gr  e47           LPA ;
     arn d5    , gr  a15   ;
     arn 1d5   , gr  a14   ;
     arn(e31)  , tl  -20   ;
     tk  20    , gr  2d    ;
     ps  e93   , vy (e4)   ;
     hv  e46           NPA ;
c=1k,tln 10    , hv  e61   ;
<i-40d6,:>; længde af end
e
```

```
b k=c, i=d6, a16 [text]
c16=d4, d4=d4+e40

a16: pm  32    D           ; k karakterinput
     pt  a6    t   a       ;
a:   hs  e20   , arn d3    ; læs eller hent symbol
     ca  19    , hv  a1    ; hop hvis t eller T
     ca  59    , hv  a15   ; hop hvis punkt
     bs (2d3)  , hv  a     ; hop hvis ≠ 0-9
     gm  d     , it  9     ;
     bs (4d3)  , hv  a     ;
     hsn e24   X           ; læs karakter nr
     tk  30                ;
     pm  d     , hs  a5    ;
e54: hh  a     , ud  a16   ; , t textinput
     pt  a6    Vt  a11     ;
a1:  bs (e23)  , it  1     ;
a9:  pa  a8    , pp  e82   ; output-case:= lower, text værdi i p
     ps  i     , hh  e20   ; læs et symbol
     bs (e23)  , hv  a6    ; hvis input-case = upper, hop
a8:  bs[output case],hh 1i ; hvis output-case = upper, hop
     hh  a7    , pa  a8    ; , output-case:= lower
     arn 58    D           ;
a11: hh  a10   , ud  i-1   ; , afslutning
     bs (a8)   , hs  a5    ; hvis output-case = upper, output lower
a15: ps  r-1   , arn a13   ; fyld sidste ord op
     hv  a5            NOA ;
     hs  a5    , qq  e4-1  ;
     hhn e91               ;

a6:  ca  5     , hh  a11   ; hvis ; , hop
     bs (a8)   , hh  a7    ; hvis out-case = upper, hop
     arn 60    D           ;
a10: ga  a8    , hs  a5    ;
a7:  arn d3    , ps  a9    ;
a5:e122:ar a12 , ca  64    ;
     ar  -1    D           ;
     cl  36    V       LOA ;
     ck  10    , cl  -6    ;
e119:hs  e57           LOA ;
     cln 7             LOA ;
     hr  s1            IOA ;

a12: 64//15                ;
a13:e59:
c=1k, qq 59.39+10          ;
<i-40d6,:>; længde af text
e

b k=e69, i=e38
i=e87, ps r-1  , arn e59   ;
     hv  e122          NOA ;
     hs  e122              ;
e
```

```
b k=c, i=e38, a8 [fejl]
c17=d4, d4=d4+e40

     can s-1   , hv  a5    ; sumfejl
a1:a2:gi a     , gk  r     ;
     pi        , sy  58    ;
     pi (a1)   IK  t-8     ;
     ncn p-1   , it  3     ;
     pa  e2    , sy  29    ;
     arn s-c2+2D   IK      ;
     ga  a2    , gs  a3    ;
     sy        , hs  e79   ; tryk fejltype
     arn d5    , hs  e81   ; tryk adresse
a:   pi        , sy        ;
     can(e2)   , hv  a4    ; hop skrivemaskine-input
[610]bs (e23)  , sy  60    ;
     sy (d3)   , sy  62    ; tryk sidste læste symbol
a3:  ps        , pm  1d    ;
     pa  e92   t 1e  93    ;
     bs  s-c8  , arn 1a1   ;
     bs  s-c5  , hr  e15   ; udhop for type 6, 7, 8 og 9
     bs  s-c4  , hr  6e4   ; udhop for type 5
     bs  s-c3  , it  b1    ; sæt udhop for type 4
     ps        t e4  -1    ; sæt udhop for type 1, 2 og 3
     pp  248   , hh  e20   ; skip til ,/: el VR
a4:  pa  e23   t -10       ; sæt lower case hvis
     hh  a3-1              ;  skrivemaskineinput

         [sumfejl]
a5:  hs  e84               ; output text sum
     qq  a6    , arn 2d3   ; hengt den korrekte sum til R
     ck  3     , sy        ; tryk bitmønster
     pa  a7    t   6       ;
a8:  sy  1     V       LO  ;
     sy  16                ;
a7:  bt  6     t   -1      ;
     ck  1     , hv  a8    ;
     pa  e2    t   1       ;
     sy  62    , pm  1d    ;
     ps (4d3)  , hh  e20   ;

a6: c=1k, k 64, 29, 58 tsum;.
<i-40e38,:>; længde af fejl
e
```

```
b k=c, i=e38, a8 [sum-check]
c10=d4, d4=d4+e40

     bs (2d3)  , hv  a     ; hop hvis clear
     gm  1d    , arn e96   ; gem M, hent sum
     tk  -5    , ar  e96   ;
     mb  a8    , ar  a8    ;
     ga  2d3   , pm  1e20  ; sum rest til 2d3
     arn a1    , gr  1e20  ; ret ord i 1e20
a:   pa  e96   , hh  e20   ; sum:= 0, læs
a1:  hh  a1    , gm  1e20  ; tilbage fra 1e20, retabler 1e20
     ca  64    , ar  a2    ;
     nc (2d3)  , hv  a4    ; hop hvis sumfejl
     pm  1d    , hh  e20   ;
a4:  gs  4d3   , ps  1     ;
     hv  1e121             ; hent sumfejlkanal og hop

a8:  qq  31                ;
a2:  qq  1                 ;
a7=i-e38                   ;

b k=c, i=d6, a1 [sæt work]
i=a7+d6

e86: vk (e46)  , vk (b11)  ;
     it (b11)  t   512     ;
     bs  448   , sk  e38   ; skriv sporingskanal
     vk  960   , vk  d21   ;
     lk  e38   , hs  e10   ; læs katalogkanal, læs antal
     ga  a     , hs  e10   ; læs beg kanalnr
     ck  -16   , ar  a     ; pak work ord
     ck  -14   , vk  960   ;
     vk  d21   , sr  3e38  ; beregn ændring af work
     ac  3e38  , sc  39e38 ; sæt work og sum
     pa  b11   t   960     ; indicer e38 tom
     sk  e38   , hv  e4    ; skriv katalogkanal
a1=e98-960                 ;
a:   c=1k, qq 4d32.31+a1.15-a1.19

<i-39d6,:>; længde af sumcheck og sæt work
[sumcelle]
e
e
c13=c17-c10, c18=d4
<d35, c31=c-c30, c=c31-c39, c=c+c+c+c+c, c=c+c+c+c, e67=c+c-1
     qq d6+40.19+402c18.40
× c31=6c39
     qq c31.39-1.39
i=c30, c:; galt antal kanaler
>
e [slip]
c30=c30-d1
```

```
b k=d42, i=0, a6 [load slip]
i=d2

<d35     [buffermode]
     vk  d50   , vk  d1+c39; dan checksum af løse kanaler
a:   lkn d49   , pp        ;
     vk  d1+c39t   1       ;
     it (a1)   , pa  a2    ;
a1:  nt  d49   ,lkd49+40d49;
a2:  ar  p     , pp  p1    ;
     ar  2     D       LA  ;
     ar  1     D       LB  ;
     bs  p472  , hv  a2    ;
     bt  c31-1c39 t -1     ;
     hh  a                 ;
     pp  p-1   , sc (a2)   ;
     vk  d1+c31-1, pp      ;
     sk (a2)   , vk  960   ;

<d39, <d35-1, <-d35+3; beregn antal karuselblokke
     pmn c31.7+c31+63 D
     dl 64 D
x <d39, <d35; beregn antal disc eller baand blokke
     pmn 9c31 D
     dl 10 D
> <d39, <d35, gr a3>
<d39, hs 1
     hv  a4
<d36,
tres;
qqf c31.39
x<d39,
     tset;                 ; set slip in cat
     qqf d35.39
a3:  qqf c31.39
[STOP, SUM]a i base of slip
s [STOP, CLEAR] > <d39
     qq 39,
     qq 57,
     qq 52,
     qqf
     tslip;
     qq c39+c38.19+d20.29+c30.39
     qqf,
a4:  hs 1
     hv  a5
     tmove;
d=d19-960
     qq 50,
     qqf c31.23+d1.39+d.29-d.33
     tslip;
     qqf,
a5:  hs 1
     hv a6
     tsetsum;
     tslip;
     qqf,
d2=i, a6: hsf2
```

```
× d2=i, hv d47
i=d48
d=d50-960, qq d35.2+d36.5+11.7+c31.23+d1.39+d.29-d.33,
     qq,
     tslip;
     qqf c39+c38.19+d20.29+c30.39
d48=i, qqf
> d1=d1+c31
e

[STOP, SUM]a i slip
s
```

```
[STOP, CLEAR]
[23.6.67                        (2)                  start, page 1]

b k=d1, i=40d13, a5, b5     ; begin start

b=80d13                     ;
<d35-2, us(-31) t - 96 >    ; ENTRY START: if aux kind = tape then rewind tape;
     vk  d19   , vk  d16    ;   read image track 0;
     lk  b     , vk  d16    ;   core init:= full init;
     pa  8b    , sk  b      ;   write image track 0;
     vk  25d16 , lk  b      ;   read image track 25;
     vk  25d16 , grn 23b    ;   clear core inhibit;
     sk  b                  ;   write image track 25;
     ps  40    , pm  b3     ;   for i:= 40 step -1 until 1 do
a:   gm  s39b          MB   ;
     ps  s-1   , bs  s      ;   buf[i]:= hsf 2;
     hv  a                  ;

a2h: pp  d13-1 , hs  c52    ; NEXT PARAM: get param;
     hv  a1    , hv  c58    ;   if number then goto set date; if single then alarm;
     hv  -9    , tl  -7     ;   if end list then return to help;
     nc  0     , hv  c58    ;   if kind ≠ drum then param alarm;
     tl  5 , ca 1.5+d32.7   ;   if work then
     arn d14+d45, tl -2     ;   R:= work-as-output;
     tl  -30   , tln 16     ;
     gr  b2    , tln -23    ;   blocks:= bits 8:23;
     hs  c3                 ;   select track (bits 24:39);
a3:  sk  40b   , arn b2     ; rep: write to drum(buf)
     sr  b1    , gr  b2     ;   blocks := blocks - 1; marks:= 11;
     hh  a2           LZ    ;   if blocks = 0 then goto next param;
     ps  a3-1  , hv  c16    ;   select next track; goto rep;

a1:  arn 2c18  D       IZA  ; SET DATE: ZA:= 0;
     hs  c2                 ;   get first track of catalog;
     grn d14+1d51, arn p1   ;   run:= bits 0:9 of param
     ga d14+1d51, sr (r)    ;
     gr d14+d51, it 4.2     ;   date:= qq 4.2 + bits 10:39 of param;
     pa d14+d51, hs  c8     ;   correct sum;
     sk  d14   , hh  a2     ;   write track back to drum; goto next param;

b1:  m  1c                  ;   count
b2:  qq [no of tracks]      ;
b3:  hsf 2                  ;
```

```
i=i+39, d=k-d1             ; d = no of tracks
b k=d42, i=0, a10          ;
a1=d19-960                 ; a1 = group no for image
a=d, <d35, a=1>            ; a = no of blocks

<d39                       ; if aux only then
i=d2, hs1                  ; begin
     hv  a4                ;   (if aux reserved then
<d36,
     tres;                 ;
     qqf a.39              ;   res, no of blocks,
× <d39                     ;   else
     tset;                 ;   set, aux kind, no of blocks,
     qqf d35.39            ;   type in)
     qqf a.39              ;
[after i follows STOP, SUM and a sum character]
ia base, start
s
[STOP, CLEAR]
>< d39                     ;
     qq  39,               ; concat pid 0,
     qq  57,               ;
     qq  52,               ;
     qqf                   ;
     tstart;               ; start, spec <
     qqf,                  ;

a4:  hs  1                 ;
     hv  a5                ;
     tmove;                ; move, bloadplace, start <
     qq  50,               ;
     qqf d.23+d1.39+a1.29-a1.33 ;
     tstart;               ;
     qqf,                  ;

a5:  hs  1                 ;
     hv  a6                ;
     tsetsum;              ;
     qqf,                  ;
d2=i                       ; end else
a6:  hsf 2                 ;
×                          ;
i=d48                      ;
     qq d35.2+11.7+d36.5+d.23+d1.39, ; load to primitive catalog;
     qq,                   ;
     tstart;               ;
d48=i, qqf                 ;
>                          ;
d1=d+d1                    ;
e                          ; end image load
e                          ; end start
[after i follows STOP, SUM and a sumcharacter]
ia start
s
```

```
;slip<
[3.7.67]                    [system punch/special binin]                    [page 1]
[STOP,CLEAR]

b i=10,b50,c20,d10,e10
     hv rc15               ; program start
b a20                      ; special binin block
d e0=552,e1=41e0,e2=41e1   ; lower buf, upper buf, save catalog

c0:  pmnra1    , gm s1     ; reestablish primitive input;
a:   ar r0     Vt 1 IQC    ;
a1:  gm s3     t -1 M      ;
     ar 2      D    LA     ;
     bs(ra)    , hv ra     ; R:= checksum;
a2:  tk 1      , ps (10)   ; prep sum error: restore s;
     vy 17     , gk rb19   ; select(t,w), save tk;
     pa 17     t 4         ; prepare SUM message;
     hv 17          NZ     ; if R≠0 then goto write SUM;
d0:  vy[d18+d17], girb19   ; save indicator;
d4:  vk[d21]   , lk e2     ; save date and run no;
     ly rb     , ly rb     ; SKIP CHECKSUM:
[-1] ly rb     , ly rb     ; for i:= lyn,lyn,lyn,lyn while i≠<aa> do;
     nc 13[aa] , hh r-1    ; INCHAR:
a3:b: lyn[char] D          ; Raddr:=lyn;
     ac (rb1)  DV   NT     ; if Raddr<0 then goto error else
     hv ra10               ;    sum1:= sum1+Raddr;
     qq (rb2)  t 1         ; sum2:= sum2+1;
     ca 66     , hv ra11   ; if Raddr=66 then goto FINISH TAPE;
     pp 39     , ca 40     ; if Raddr=40 then p:=-1 else
     pp -1     V           ;    if Raddr=1 then p:=39 else
     nc 1      , hv ra10   ;    goto error;

a4:  pmn 1.3   DX   IZA    ; INWORD: RM:=0; LZA:=true;
a5:  tl  -7    , ly r1     ; for i:=1 step 1 until 6 do begin
[1]  pi [marks]t 12 LZA    ;   RM:=RM:128 ∨lyn; if LZA then begin
b1:  ac [sum1] DXV  NT     ;   RC:=bits 8 and 9; LZA:=false end;
     hv ra10               ;   if Raddr<0 then goto error else
     hv ra5    X    LZ     ;     sum1:=sum1+Raddr  end;
b2:  qq [sum2] Xt 6        ;   sum2:=sum2+6; p:=p+1;
     tl 3      , pp p1     ; cell[buf+p]MRC:=RM×8;
b3:  gr pe0         MRC    ;
     bs p473   , hv ra4    ; if p<39 then goto INWORD;
     bs p-39   , hv ra9    ; if p=40 then goto AREAWORD;
     pm rb6    V    NQA    ; NEXT TRACK:
     arn rb10  , hv ra7    ; if disc then goto outtrack;
     dlnrb7    , ar rb7    ; vk(track:960+960);
     ck -10    , ga rb4    ;
b4:  vk [group], cln -20   ;
b5:  gt rb5    , vk [track]; vk(track mod 960);
     is (rb3)  , sk s-39   ; write track from(buf);
a6:  arn rb8   , ac rb6    ; track:=track+1;
                           ;
     nt -41    , it 0      ; buf:= if buf=lower then upper else lower;
     qq (rb3)  , hv ra3    ; goto INCHAR;

b6:  qq [current track]    ;
b7:  qq 960.39             ;
b8:  qq 1.39               ;
```

```
d1:b9: qq [d53.9]+0.39     ;  parameterwords for
[1d9]  qq [d53.9+53.39]   ; us [unit]

b10: qq e0.9+40.19-40.39   ; parameterwords for
[1b10] qq e0.9+40.19-40.39 ; us 0

b11: qq 40.39              ;
[1b11] qq 1.21             ;
                           ;outtrack:
a7:  ar rb11   , us        ; param:=param+40; us(0,buf,param)
d2:b12:                    ; count:=count+1;
     bt [count] t -1       ; if count≥0 then goto INCHAR;
     gr rb10   , hv ra3    ;
d3:b14:                    ;outblock:
     pp [d53:40-1], grrb10 ; count:=block:40-1;
b13: can -1    , hs ra8    ; if -,first then sense;
b17: nt 1b9-i  , arnrb9+1b9; exchange buffers;
     ar rb6    , us (rb18) ; us(unit,track,buffer);
     nt 1b10-i , pmrb10+1b10;
     gm rb10   , pa rb13    ; param:=new buffer base-40;
     arnr1b11  , ac rb6    ; track:=track+1;
     gp rb12   , hv ra3    ; goto INCHAR;

b15: qq                    ; status word
[1b15] qq b15.9+1.19+4095.39; get status

                            ; procedure sense;
a8:  pa rb16   , arnr1b15   ;   begin rep:=0;
b18: il [unit] , il        ; L:  if status word(unit)<0 then
     arnrb15   , pm rb6    ;     begin
     hv s1          NT     ;     rep:=rep+1;
b16: bt [rep]  Xt -150     ;     if rep>3 then
     hv ra10               ;       goto error;
     pa r1     , it (rb17) ;     us(unit,track,buffer);
[1]  is r0     , ar si-b17 ;     goto L
     us (rb18) , hh ra8    ;     end
                           ;   end;

a9:  hs ra8         LQA    ;AREAWORD: if disc then sense;
     pm (rb3)  X    IQC    ; R:=set marks(cell[buf+p]); disc:=LA;
     pa rb3    Vte0   LQA  ; cat:=LB; if -,disc then begin
     gr rb6    , hv ra3    ; tracks:=R; goto INCHAR end;
     ga rb18   , tk 18     ; unit:= part1(R)+16; track:=Rpos21;
     it (rb18) t 16        ;
     pa rb13   , gr rb6    ; first:=true;
     vk 960    , hv ra3    ; wait for track; goto INCHAR;
```

```
a10:
b19: pi [in]   , vk [tk]   ;error: restore indicator; restore tk;
     arnr      , hv ra2    ; R:=not zero; goto prep sum error;

a11: lynrb     , tk -7     ;FINISH TAPE:
     ly rb     , tk -7     ; R:=((lynshift-7 vlyn)shift-7
     ly rb     , tk 4      ;    vlyn)shift 4;
     nc (rb2)  , hv ra10   ; if part1(R)+sum2 v
     ck 10     , vk 960    ;   part2(R)+sum1 then
     nc (rb1)  , hv ra10   ;   goto error; select group(0);

     vk (rd4)  , ps (10)   ; restore s;
     hv ra13        NQB    ; if cat then begin
     lk e0     , vk (rd4)  ;   fetch first catalog track;
     pp 5      , pm pe0    ;
     pp p1          LB     ;   p:=rel address of date;
     pm pe2    , gm pe0    ;   date:=saved date;
     pm p1e2   , gm p1e0   ;   run no:=saved run no;
     pm 1      DX          ;
a12: ar e0-1    t 1         ;
     ar 2      D    LA     ;
     ar 1      D    LB     ;   form new checksum;
     sc 39e0   V    LC     ;
     hv ra12               ;
     sk e0     , vk 960    ;   write track end;

a13: arn 35    , ca 37     ; if no summation then
d5:  ps [c45-c41], hh -9   ;   goto SET TYPEWRITER;
     vk 1      , lk 40     ; fetch track 1;
     pi (rb19) , ud rb19   ; restore indicator; restore track;
     qq (17)   t 1         ; prepare track 1 call of SUM;
     arnrb19   , tk 10     ; if catalog track was selected then
d6:  ca (rd4)  , hv [c53]  ;   goto read and check Help;
     hv 39                 ; goto end of track 0;

c1:  qq f                  ; checksum word;

e                          ; end special binin;
```

```
b a20
d e7=552                    ; binout buf

a0:  pp e7      , pmn p     ;OUTPUT:
     ar 1.1     D     LA     ; for i:=words step 1 until 1 do
     ar 1.2     D     LB     ;   begin
b20: qq [sum2]Xt 6           ;   sum2:=sum2+6;
a2:  cl 32      , ga r1      ;   R:=outbuf[p];
[1]  sy -1      , it -100    ;   M:=marks(outbuf[p]);
b21: bt 0       , hv ra1     ;   for j:=1 step 1 until 6 do
     cl -7      , hh ra2     ;     writechar(char(j)of:(RM));
a1:  pa rb21    , pp p1      ;   p:=p+1;
b22: can[words] t -1         ;   end;
     hr s1                   ; return;
a3:  hh ra0                  ;

[Bootstrap program, 6 7-bit characters / word.      Bits of: Curr instr, prec.]
b23:   qq   2.6+ 0.13+ 0.20+  0.27+ 0.34+ 0.39  ;hv s             25-39, length
               [bits 29-34, 36-38 is length]
       qq   2.6+ 4.13+ 0.20+  0.27+ 0.34+ 7.39  ; gm s7 t-1 M     25-39, 0-24
       qqf 28.6+ 2.13+ 0.20+127.27+63.34+11.39  ; hv s     IKC    19-39, 0-24
       qq  32.6+ 6.13+56.20+ 64.27+ 0.34+ 0.39  ; gr xx t1 MPC    25-39, 0-18
[4b10] qq   0.6+ 0.13+ 0.20+  0.27+64.34+10.39, ; pi 0 t -49      31-39, 0-24
               [bits 15-20, 22-24 is 510 - length]
       qq  28.6+ 0.13+ 7.20+103.27+49.34+ 8.39  ; hv xx    MR     19-39, 0-30
[6b10] qq   0.6+32.13+96.20+  0.27+ 0.34+ 0.39  ; tl 12 V  IK     25-39, 0-18
               [bits 22-27, 29-31 is 513 - length]
       qqf 24.6+44.13+ 1.20+ 32.27+64.34+ 2.39  ; bs (r4)  IO     19-39, 0-24
       qq   0.6+ 0.13+32.20+ 64.27+ 0.34+ 0.39  ;                  fill , 0-18
       qq             64.20+  0.27+ 0.34+ 0.39  ;                  fill

b24:   qq 45.39              ; length of track 0
[1b24] qq c1.39-c0.39+1.39   ; length of special binin

c2:    vy 32     , sy 14     ;binout track 0 form:
a5:    sy 17     , pt -1     ; writetext(‡≤‡); sum1:=0;
       pp -1     , pp p1     ;
       pa r1     , it rb23   ; fetch boot;
[1]    pm p0           IPC   ;
       gm pe7          MPC   ;
       bs p503   , hh r-4    ;
b25:   pmn rb24  , tl 36     ;
       tk 5      , ac e7     ; pack(length)into:(boot[0]);
       tln 4     , ac e7     ;
       pa ra6    t   507     ;
a6:    arn 507   Dt 3        ;
       ck 10     , sr (rb25) ;
       tl -3     , tk 19     ; pack(510-length,boot[4]);
       ac 4e7    , tln 18    ;
       ac 4e7    , ud ra6    ;
       ck 10     , sr(rb25)  ;
       tl -3     , tk 12     ;
       ac 6e7    , tln 11    ; pack(513-length,boot[6]);
       ac 6e7                ;
       pa rb22   t 10        ; words:=10;
       pa rb20   t 1         ; sum2:=1;(prepare final 64 mark)
       hs ra0                ; OUTPUT;
       pm(rb25)  , gm rb47   ; current length:=length;
```

```
a7:b26:                          ;OUT BIN0:
       pmn rc12  t 1             ; iword:=iword+1;
       ar 1.1    D     LA        ;
       ar 1.2    D     LB        ; RM:=(set marks(cell[iword])pos 39+
       qq(rb20)  t 7             ;   marks pos 78) shift 34;
       ck 3      X               ; sum2:=sum2+7;
       cl 34     , pp -6         ; for p:=-6 step 1 until 0 do
a8:    ck -4     , ga r2         ;  begin
       bs p1     , it 64         ;  writechar((if p=0 then 64 else 0)+
[2]    sy -1     , pp p1         ;   Raddr shift -4);
       bs p511                   ;  RM:=RM shift -6
       cln -6    , hv ra8        ;  end;

       arn rb47  , sr rb8        ; current length:= current length-1
       sy  64    V     LZ        ; if current length≠0 then
       gr rb47   , hv ra7        ;  goto OUT BIN0;
c17:   it(rb20)  , pa -1         ;WRAPUP: R:=sum2×1024+sum1;
       arn -1    , ck 10         ;
       ga r1     , ck -7         ; writechar(bits(3)throu:(9) of:(R));
       sy -1     , ga r1         ; writechar(bits(36,2,R));
       sy -1     , ck -7         ; writechar(bits(29,35,R));
       ga r-1    , sy(r-1)       ;
       vy 17     , hr s1         ; select(t,w); return;

b30:   qq 1.39                   ; heading word (initially help)
[1b30] qq[d37]                   ; length           -      -
[2b30] qq 1.39                   ; abs base

c3:                              ; procedure binout; begin
b27:   can -1    , hv ra10       ;  if no heading yet then
       pa rb26   t c0-b26-1      ;  begin binout track 0 form (special binin);
       qq(rb25)  t 1             ;
       hs rc2                    ;
       pa rb20   t 1             ;   sum2:= 1;(prepare final 66 mark)
       vy  32    , sy 13         ;   select(p); writechar(13);
       pa rb27   , pt -1         ;   no heading yet:= false; sum1:= 0 end;
a10:   gs rb28   , vy 32         ;  select(p);
       qq(rb20)  t 1             ;  sum2:= sum2 + 1;
       pm rb30         IPA       ;  disc out:= A mark(heading word);
       sy 1      , pp rb30       ;  writechar(1);
       pa rb22   t 1             ;  words:=1;
       pa rd7    , hs ra3        ;  count:= 0; OUTPUT(heading word);
a11:   hh ra14         LPA       ;NEXT TRACK: if -,disc out then
       pm r2b30  , hs rc8        ;   begin select track(abs base);
       arn rb8   , ac r2b30      ;   abs base:=abs base+1;
       lk e7     , vk 960        ;   read track to(track buf);
       arn r1b30 , sr rb8        ;   length:= length-1
       gr r1b30        IZB       ;   LZB:= length=0;
a13:   pa rb22   t 40            ;OUT TRACK:
       qq(rb20)  t 1             ;   words:=40; sum2:= sum2+1;
       sy 40     , hs ra0        ;   writechar(40); OUTPUT(trackbuf);
       hv ra11         NZB       ;   if -,LZB then goto NEXT TRACK;
b28:   ps[s]     , vy 17         ;   select(t,w);
      [hr s1     ,]              ;   return end;
```

```
a14:   hr s1     , arn rb10  ;   else begin bufad:=bufad+40;
a15:   ar rb11   , il        ;   il(0,trackbuf,bufad); LZB:=false;
d7:    bt[count] t -1   IZB  ;   count:=count-1; if count≥0 then
       gr rb10   , hv ra13   ;   goto OUT TRACK;
       is(rd3)   , it s1     ;   count:=block:40;
       pa rd7                ;
       pa rb31   , pm r1b10  ;   rep:=0; bufad:= -40;
a16:   arn rb30  , ga rb29   ;repeat input: unit:= part1(heading word);
       arn r2b30 , tk 28     ;
       ck -10    , ar rd1    ;   il(unit,0,blockpart(abs base));
b29:   il[unit]  , arn r1b15 ;
       is(rb29)  , il s16    ;   if status(unit)<0 then
       il        , arn rb15  ;    begin rep:= rep+1;
b31:   bt[rep]   Vt -150 LT  ;    goto if rep>3 then ERROR
       srn rb8   , hh r2     ;     else repeat input
       pp rb33   , hh rc7    ;    end;
[+2]   hv ra16   , sc r2b30  ;   abs base:= abs base + 1;
       ar r1b30  , gr r1b30  ;   length:= length - 1; if length<0 then return;
       hv ra15   X      NT   ;   goto a15
       hv rb28               ;   end
e
b a20
d e3=-86[search base],e4=40e7[cat in],e5=40e4[cat out],e6=40e5
a:
c4:    pm p      , arn ra1   ; procedure writetext(p); integer p; begin
a2:    cl -6     , tk -4     ;next word: M:= cell[p]; R(37:39):= 3 1;
       ga ra3    , ca 15     ;next char: char:= M∧15; RM:= RM:64;
       pp p1     , hv ra     ;  if char=15 then begin p:= p+1;
a1:    ca 10     , hv s1     ;   goto next word end;
b32:   qq[pos]   t 1         ;  if char≠10 then begin
a3:    qq[char]  , ca 63     ;   pos:= pos+1; R:= sum1; writechar(if char=63
       qq(ra3)   t 1         ;   then 64 else char); sum1:= R;
       arn -1    , sy(ra3)   ;   goto next char
       gr -1     , hv ra2    ;  end end;

a4:                          ; Boolean procedure querry; begin M:= sum1;
c5:    pm -1     , lyn ra5   ;L: i:= lyn; if i=<n> then
       ca 37     , hv r4     ;  begin writechar(<o>); querry:= false
       nc 24     , hv ra4    ;  end else if i≠<y> then goto L else
       sy 53     , sy 18     ;  begin writechar(<e>); writechar(<s>);
       gm -1     , hv s1     ;  querry:= true end; sum1:= M end;
       sy 38     , gm -1     ;
a5:    qq        , hh s      ;

c6:    hs 29e3               ; integer procedure get;
       hh s      X      LZ   ;  if get word=0 then get:= M else catsum;
c7:    pp rb34   , hs rc4    ;   begin writetext(⦇catalog⦈); stop end;
       zq        , hv r      ;ERROR: writetext(⦇fault⦈); stop;

b33: k63,29
tfault;
62.
b34: k63,29
tcatalog;
62.
b35: k63,29
tfull;
62.
```

```
c8:    dln rb7   , ar rb7    ; procedure select track(M); integer M;
       ck -10    , ga r1     ;  begin
[1]    vk[group] , cln -20   ;  vk(M:960 +960);
b48:   gt r      , vk[track] ;  vk(M mod 960)
       hr s1                 ;  end;
c9:
b36:   gr e5-1   t 1   MPC   ;out: oword:= oword+1; cell[oword]:= R;
       it(rb36)  , bs 38e5   ; marks[oword]:= PC; if oword<rel 39 then
       hr s1                 ;  return;
       grn 39e5        MC    ; cell[rel 39]:= 0; marks[rel 39]:= C;
b49:   pm[reltr] DXt 1       ; reltr:= reltr+1; j:= rel 0 -1;
       pa rb36   t e5-1      ; for sum:= reltr, sum+setmarks(cell[j])
       pa r1     t e5-1      ;  while -,LC do j:= j+1;
a6:    ar -1     t 1         ;
       ar 2      D     LA    ;
       ar 1      D     LB    ;
       sc 39e5   V     LC    ; cell[rel 39]:= -sum;
       pm rb38   , hv ra6    ;
       hs rc8                ; select track(work base 1);
       sk e5     , vk 960    ; write track from(rel 0); wait for track;
       arn rb8   , ac rb38   ; work base 1:= work base 1+1;
       sc rb37         ITB   ; work length:= work length-1;
       hv s1           NTB   ; if work length≥0 then return;
       pp rb35   , hh rc7    ; writetext(‹full›); stop;

e
b a                          ;
d a=i                        ; the following locations overwrite
                             ;  the program initialization
b37:   qq[work base]         ;
b38:   qq[work base 1]       ;
[1b38] qq[work base 2]       ;
[2b38] qq[first free]        ;
[3b38] qq[old length]        ;
b47:   qq[current length]    ; used in out bin 0
d i=a
e
```

```
b a30
c15:   arn e3    , ga rd0    ;START PROGRAM: ;comment
       gt rd6    , tk 20     ; fetch and spread system parameters:
       ga rd5    , tk 10     ;  d18+d17, c53, c45-c41, d53
       ac rd1    , ac r1d1   ;  d37 is fetched at 2a1
       ck 10     , ac r1d1   ;  d53:40-1
       xr        , dln rb11  ;
       sr rb8    , ck -10    ;
       ga rd2    , ga rd3    ;
       arn 24e3  , ck -10    ;  d21;
       ga rd4    , vyn 17    ; select(t,w);
a0:    ar r1c0   t 1         ;
       ar 2      D      LA   ; form checksum for special binin;
       hv ra0           NB   ;
       sc rc1           M    ;
c10:                         ;TRACK 0:
       pp rb39   , hs rc4    ; writetext(ǂ<track 0ǂ);
       hs rc5    , hv ra1    ; if querry then begin
       vk 960    , vk 0      ;  fetch track 0;
       lk rc11   , vk 960    ; OUT BIN 0(track 0)
b42:   qq e4     , hs rc2    ;  outspace(100)
       vy 32     , pa r1     ; end;
       bt 0      t -5        ;
       vy 17     V           ;
       sy 0      , hv r-2    ;
                             ;HELP:
a1:    pp rb40   , hs rc4    ; writetext(ǂhelpǂ);
       hs rc5    , hv ra2    ; if querry then
       arn 1e3   , tl -30    ; binout(1) length: (d37);
       gr r1b30  , hs rc3    ;

a2:    arn rb42  , ga 14e3   ; assign buffer area for search;
       hs 5e3           IZA  ; fetch first catalog track;
       hv rc7           NZ   ; if R≠0 then goto catsum;

a3:    pm(29e3)  X      IPC  ;TAKE LAST ITEM: R:= set PC(store[iword]);
       hv ra21          LZ   ; comment see also Help 3 page 13;
       ga rb43               ; if R=0 then goto MAYBE END;
b43:   pi[kind]  t 560       ; in:= kind;
       hh ra6           NTB  ; if special then begin
       hh ra4           NQA  ; if free then begin
a5:    tl -32    , tln 16    ;set work:
       gr rb37   , tln 16    ;   work length:= blocks(R);
       gr rb38   , gr r1b38  ;   work base1:= work base2:= first block(R);
       arn(29e3) , hs rc9    ;   out(store[iword]);
       hv ra7           NQA  ;   if -,free then goto name comes;
       hs rc6    , mb rb44   ;   first free:= get∧24 0 16 m;
       gr r2b38  , arn(29e3) ;   R:= store[iword]∧8 m 16 0 16 m;
       ck 8      , tk 16     ;   comment booked:= 0;
       ck -24    , hs rc9    ;   out(R); goto name comes end free;
a4:    hv ra7    , hs rc9    ;work: out(R);
       hs rc6    , qq        ;  R:= get;
       hv ra5           LPA  ;  if LPA then secondary work: goto set work;
      [hh ra8    ,]          ;  goto next out end special;
```

```
a6:    hh ra8    , tl -7     ; if kind=drum
       nc 0      , ca 1      ;   ∨kind=disc then
       pp e6-1   , hh ra9    ;  goto STACK ITEM;
       tl 7      , hs rc9    ; out(R); R:= get;
a8:    hs rc6    , hs rc9    ;next out: out(R);
a7:    hs rc6    , qq        ;name comes: R:= get;
[1a7]  qq[see 1a10] V   LZ   ; goto if R≠0∧NA then
       hh ra8           NA   ;  next out else TAKE LAST ITEM;
a9:    hv ra3    , gp rb46   ;STACK ITEM:
       tl 7      , ud ra10   ; istack:=stack bottom;
       hs rc6    , qq        ; for R:= R, get, get while NA∧NA do
a10:                         ;  begin
b46:   gr[istack] t 1   MPC  ;  store[istack] MPC:= R;
       hs rc6    , ud r1a7   ;  istack:= istack+1;
       hv ra10          NA   ;  end;

       pp 1e6    , pm p      ; pos:= 0; M:= sum1; writecr;
       pp p1            LA   ; sum1:= M; in:= bits and kind;
       pm -1     , sy 64     ;
       gm -1     , pi(rb43)  ; writetext(address(stack[if Amrk(
       pa rb32   , hs rc4    ;  stack[1]) then 2 else 1]));
a11:   bs(rb32)  t 7          ; for pos:= pos+1 while pos≤7 do
       sy 0      , hv ra12   ;  writechar(0);
       qq(rb32)  t 1         ; writechar(0);
       sy 0      , hv ra11   ;
a12:   hs rc5    , hv ra17   ; if -,querry then goto NO;
       pm e6     X           ;YES: R:= stack[1]; comment the areaword;
       hv ra13          LPB  ; if -,reserved then begin
       tl -32    , tln 16    ; length:= blocks(R);
       gr r1b30  , tln 4     ; heading word:= first block(R)+
       tl 12     V      NTA  ;  (if disc area then unit(R) else 0) pos 9+
       ck 18     , tl 12     ;  (if disc area then 1 else 0) pos 40;
       gr rb30          MTA  ;
       mb rb44   , gr r2b30  ; abs base:= first block(R);
a16:   ps ra19   , hv rc3    ; binout; goto OUT STACK
                             ; end else begin
a13:   pm r2b38  , mb rb44   ; heading word:=
       gm rb30          MTA  ;  if disc area then
       gr r2b30  , tk  18    ;
       hv ra14          LTA  ;  first free-unit(first free)
       tl 23                 ;  +unit(stack[1]) pos 9+1 pos 40
       arn e6    , ck -16    ;  else first free;
       tl 16     , hv ra15   ; abs base:= stack[1]∧24 0 16 m;
a14:   ga rb30               ; stack[1]:= stack[1]∧24 m+first free;
       tl 27     , arn e6    ;
       ck -12    , tl 12     ; length:= blocks(stack[1]);
a15:   gr e6     , ck 8      ; first free:= first free+length;
       tl -24    , gr r1b30  ;
       ac r2b38  , hv ra16   ; binout; goto OUT STACK end YES;
```

```
a17:   hv ra3          LPB  ;NO: if reserved then goto TAKE LAST ITEM;
d a19=i-1                   ;OUT STACK:
a18:   pa rb45   t e6-1     ; for j:= stack bottom
b45:   pm[j]     Xt 1   IPC ;  step 1 until istack do
       hs rc9               ;  out(store[j]);
       pm rb45   X          ;
       nc(rb46)  , hv rb45  ;
       hv ra3               ; goto TAKE LAST ITEM;

a21:   pa r2     Vt 39  LB  ;MAYBE END: if -,LB then
       hs rc6    , hv ra3   ; begin get; goto TAKE LAST ITEM end;
                            ;END SCAN:
[-1]   bt 39     t -1        ; for j:= 1 step 1 until 39 do
       ps r-2    , hvn rc9  ;  out(0);
       pm r1b38  , hs rc8   ; select track(work base2);
       lk e4     , ud rb48  ; read track;
       arn e4    , ck 2     ; R:= free areaword;
       pm e4     X      IOB ;
       ck 8      , tl -24   ; old length:= blocks(R);
       gr r3b38             ; free areaword:=
       tln -23   XV     NOB ; (if free kind=drum then
       tln 4     , hh ra22  ;  (first block(R)+old length
       ar r3b38  , sr r2b38 ;  -first free)pos23+first free
       tk 16     , ar r2b38 ;  else
a22:   hv ra23   , arn r2b38;  (first block(R)+old length
                            ;  -(first free∧28 0 12 m))pos23+first free
       sc r3b38  X          ;  +free kind pos2)
       tl -27    , ar r3b38 ;
       tk 16     , ac r2b38 ;  +special+booked to be set;
      pm e4     , tl 23    ;
      arn r2b38 , ck -16   ;
      tl 4      , ck 12    ;
       it 1.2               ;
a23:   ar 1.3+1.6 D         ;
       gr e4     , it 1     ; reltrack in search:= 1;
       pa 15e3   , hs 15e3  ; sum track;
                 , ud rb48  ;
       sk e4     , vk 960   ; write track; wait for track
       pm 24e3   , arn rb49 ; heading word:= d21 pos39+1 pos41;
       gm rb30          MB  ; abs base:= work base 2;
       pm r1b38  , gm r2b30 ; length:= reltr pos 39;
       tl -30    , gr r1b30 ;
       pp rb41   , hs rc4   ; writetext(‡catalog‡);
       hs rc5    , hv r2    ; if querry then
       hs rc3               ;  binout;
       vy 32     , sy 66    ; select(p); writechar(66);
       hs rc17              ; WRAPUP;

       pa rb25   t b24-b25  ;PREPARE NEXT RUN:
       pa rb26   t c13      ; prepare track 0 output;
       pm r1b10  , gm rb10  ;
       grnrb15          M   ;
       pm rb8    , pa rb49  ;
       pa rb36   t e5-1     ; prepare out;
       gm rb30          M   ; prepare help output;
       gm r2b30  , it -1    ; no heading yet:= true;
       pa rb27   , hv rc10  ; goto TRACK 0;
```

```
b44:   qq 1.23-1.39          ; mask 24 0 16 m;
b39: k58,62,63
t
track 0  ;
.
b40: t
help     ;
b41: t
catalog  ;

d c12=i-1,c13=c12-b26

       vk 960    , vk 0      ; track 0 heading
       ly r      , ly r      ;
       ly r      , sk rc11   ;
       lk 0      , vk 0      ;
       hv 1                  ;
c11:
e                            ; end program block

c16:   hs 1                  ;
       hsf 2                 ;
       tbinout;              ;
       qq 16     ,           ; binout, 0 10..c11<
       qqf 10.19+c11.39      ;
       qqf       ,           ;

[STOP,SUM ]P
isystem punch

ec16
```

```
;slip<
[29.7.67                              (2)                    Check bin, page 1]
[STOP, CLEAR]
b a15, b15
i=10

a4:  vy  17    , pa  rb2   ; START: select(t, w); sum:= 0;
     pp  rb    , hs  ra6   ;   message(¢<check bin¢});
b1:  pa[chars] D           ;   chars:= 0;
     ly  ra    , vy  16    ;   typechar; select(r, w);
a:   lyn       DV          ; search begin: read symbol;
     pp  rb3   , hv  ra5   ;   if parity error then alarm(¢<parity¢});
     ca  13    , hv  ra1   ;   if aa then goto binin form;
a2:  nc  14    , hv  ra    ;
     lyn ra    V           ;   if not ≤ then goto search begin;
    pp  rb3   , hv  ra5   ;
     nc  17    , hv  ra2   ;

     pp  1     , hsn ra7   ; BIN 0 FORM:
     pp  1                 ;
     tk  1     , hs  ra7   ;
     tk  3     , ga  rb4   ;   words:= read(1), read(1);
     pp  59    , hs  ra7   ;   read(59); skips bootstrap + 1 char;
     pp  7                 ;   for i:= words step -1 until 1 do
b4:  bt[words] t   -1      ;
     ps  r-3   , hv  ra7   ;   read(7);
     hv  ra8               ;   goto check sum;
a1:
a1h: pp  1     , hs  ra7   ; BININ FORM: read head symbol;
     ga  rb4   , it  63    ;   if symbol ≤ 63 then
     bs(rb4)   , hv  ra3   ;   begin
     pp  6                 ;     for i:= symbol step -1 until 1 do
     bt (rb4)  t   -1      ;     read(6);
     ps  r-3   , hv  ra7   ;     goto binin form;
     hv  ra1               ;   end;
a3:  ca  66    , hv  ra8   ;   if symbol = end then goto check sum;
     nc  64    , ca  65    ;   if symbol = repeat  ∨ symbol = label then
     pp  4     , hh  ra1   ;   begin read(3); goto binin form end;
     pp  rb5   , hv  ra5   ;   alarm(¢<tape sum¢});

a7:  it  p     , qq (rb1)  ; procedure read(p); begin chars:= chars + p;
a9:  tl  -7    , pp  p-1   ;   for p:= p step -1 until 1 do begin
     ly  ra    V           ;   RM:= (RM shift -7) + lyn;
     pp  rb3   , hv  ra5   ;   if parity error then alarm(¢<parity¢});
b2:  ac [sum]  D           ;   sum:= sum + char;
     bs  p     , hv  ra9   ;   end;
     hr  s1                ;   end read;

a8:  it (rb2)  , pt  rb6   ; CHECK SUM: saved sum:= sum;
     pp  3     , hs  ra9   ;   read(3);
     tk  4     , ca (rb1)  ;   if chars ≠ first part
b6h: ck  10    , nc  -1    ;      ∨ saved sum ≠ second part then
     pp  rb5   , hv  ra5   ;     alarm(¢<tape sum¢});
     hv  ra4               ;   goto start;
```

```
a5:  sy  29    , ps  ra4-1 ; ALARM: writered; prepare return to start;
a6:  sy  64    , sy  58    ; MESSAGE: writecr;
a11: pmn p     X           ;   p points to text;
a10h:cl  34    , ck  -4    ; rep:
     ga  rb7   , ca  15    ;   Raddr:= next char;
     pp  p1    , hv  ra11  ;   if Raddr = 15 then get next word;
     ca  10                ;   if Raddr = end text then
     sy  62    , hr  s1    ;   begin writeblock; return end;
b7:  sy  -1    , cl  -6    ;   write char;
     hh  ra10              ;   goto rep;

b:   tcheck bin;           ;
b3:  tparity;              ;
b5:  ttape sum;            ;

a12: hs  1                 ; ENTRY AFTER SLIP LOADING:
     hsf 2                 ;
     tbinout;              ; binout, 0 first ..top<
     qq 16,                ;
     qqf a4.19+a12.39      ;
     qqf,                  ;

[STOP, SUM]M i check bin
e a12
```

```
;slip<
[4.7.67                                              Cattap, page 1]
[STOP, CLEAR]

b a11                        ;
i=483, a6=512                ; check last in cell 512
a11: vy  17    , sy  64      ; START:
     sy  51    , sy  49      ;   writetext(↯<cattap↯);
     sy  19    , sy  19      ;
     sy  49    , sy  39      ;
     ly  a     , pp (a)      ;   p:= station:= typechar;
a:   ly  -1    D             ;   if typechar ≠ comma then
     nc  27    , hv  a11     ;   goto start;
     arn a1    , us  0       ;   buffer[1]:= ↯<cattap↯;
     usn p64   , arn a2      ;   rewind;
     us  p     , iln p160    ;   writelabel;
     arn a3    , il  0       ;
     arn a4                  ;   if error then
     hv  a5              NT  ;   begin
a7:  sy  64    , sy  54      ; write error:
     sy  49    , sy  20      ;     writetext(↯<fault↯);
     sy  35    , sy  19      ;     goto start
     hv  a11                 ;   end;
a5:  arn a6    , us  0       ;   buffer[1]:= EOF;
     arn a2    , us  p144    ;   write EOF even parity;
     iln p160  , arn a3      ;
     il  0     , arn a4      ;
     hv  a11             NT  ;   if error then goto write error;
     hv  a7                  ;   goto start;

a8:  tcattap;                ;
a1:  qq a8.9+1.19+1.39       ;
a9:  qq 15.5+15.11+15.17+15.23 ;
a2:  qq      1.19+1.39       ;
a4:  qq                      ;
a3:  qq a4.9+1.19+0.39       ;
a6:  qq a9.9+1.19+1.39       ;

a10: hs  1                   ; ENTRY AFTER SLIP LOADING:
     hsf 2                   ;
     tbinout;                ;   binout, 0 first..last<
     qq 16,                  ;
     qqf a11.19+a6.39        ;
     qqf,                    ;

[STOP, SUM]Ai cattap
e a10
```

```
;30.6.67.  Punch head bin0      PM.

bi=10,a5                  ;
    arn a3     t-1        ;skift de første 4 ord som indlæses med kanal 0
    tk  3      ,gr (i-1)  ;3 skift til venstre.
    bt  3      t-1        ;
    vy  32     ,hv i-3    ;og vælg strimmeloutput
a2: pmn a      IRC t-1    ;et ord til M
    ar  256    D LRA      ;Amærke
    ar  128    D LRB      ;og Bmærke i R
    xr         ,tl -10    ;
    ar  128    D          ;sæt ord slut mærke
a4: qq  440    t-1        ;tæl karakterer
    tl  -7     X IZA      ;næste karakter i tæledel
    gt  i      ,sy        ;hul næse karakter
    hv (a4)    XVD NZA t-1;hop med karaktertælling hvis flere karakterer i dette ord
    bs (a2)    ta1        ;hvis flere ord
    hv  a2     ,sy  49    ;hop til a2
    bt (a4)    t-2        ;fyld op med mellemslag til ca 435 karakterer
    sy  6      ,hh i-2    ;
    ly  a5                ;læs en karakter
    nc  14     ,hv  i-1   ;skip indtil understregning
a5: sy         ,ly  i     ;kopier
    nc  11     ,hv  a5    ;indtil stopcode
    sy  11                ;hul stopcode
    bt  100    t-1        ;
    sy         ,hv  r-1   ;hul ca 100 mellemslag
    hsf 2                 ;hop ud til hjælp

a1: ly         D          ;det følgende læses med gammel kanal 0
    nc  11     ,hv  r-1   ;skip indtil stopcode
    qqn                   ;
    tl -6      ,ca  0     ;primitivt læseprogram
    ly  r4     ,hs  r-1   ;
    gm  s3     M t-1      ;

    tl  3                 ;boot
    gr  r-a1+2 MRC t-1    ;
    hv  34                ;
    gr  42     MRC t-1    ;
a3: 272/351/ 68/  3a      ;
a: e10

det følgende kopieres til strimmel
≤42     2æ
i=10,vyn16
tl-6,ca
lyr4,hsr-1
gms3t-1M
e10
```

```
;slip<
[Punch head kompud.   PM 1.7.68 version 6
SET TAB|             |             |clear code ]

b b16,d,e88                                                    ;
d=1                                                        ;
i[reefine
;sum code
s;clear code ]                                               ;
d=d+d+d+d+d, d=d+d+d+d, d=d+d+d+d, d=d+d+d+d, e88=d-320; d=antal kanaler

b a7                      ;
i=10, vy 32   ,pp         ;punch boot
      sy 14   ,sy 17      ;≤
a2:   pmn b   X t-1       ;næste ord
      bs (a2) tb16        ;
      cl 1                ;
a1:   cl 36   ,tk -4      ;næste char i R0-9
      bs pb1  tb2         ;
      qq      V           ;
      bs pb3  tb4         ;
      ar 64   D           ;sæt ordmærke
      ga a    ,cl -6      ;
a:    sy      ,bs pb5     ;hul char
      pp pb6  ,hh a1      ;tæl char
      bs (a2) tb8         ;
      pp pb10 ,hv a2      ;tæl ord

a3:   qq 6    ,pt -1      ;punch kompind
a4:   pmn b8  IRC t-1     ;næste ord
      ar 256  D LRA       ;A-mærke
      ar 128  D LRB       ;B-mærke
      xr      ,tl -10     ;
      ar 128  D           ;
      tl -7   X IZA       ;
      gt r    ,sy         ;
      hv r-2  X NZA       ;
      bs (a4) tb11        ;
      hv (a3) VD t6       ;
      it (a3) ,pa -1      ;
      arn -1  ,gt a5      ;checksum og antal
a5:   tk -7   ,sy         ;
a6:   gt a6   ,sy         ;
      tk 3    ,ga a7      ;
a7:   sy                  ;
      bt 100  t-1         ;mellemslag
      sy      ,hv r-1     ;
      vy 17   ,ly r3      ;vent
      nc 51   ,hsf 2      ;
      vy 32   ,ly r1      ;stop ,kopier
      sy      ,hh r-1     ;
e                         ;hullesekvens slut

b a23, c6                 ;kompind som hulles
b11:  lyn ra9 ,tk -7      ;læs checksum og antal for kompind
      ly ra9  ,tk -7      ;
      ly ra9  ,tk 4       ;
      gs ra21 ,ca (sb15)  ;
      tk 10   ,nc (sb12)  ;
      hv ra21             ;sumfejl i kompind
      ly ra9              ;skip til <
      nc 17   ,hv r-1     ;
e:    pa a23  t51         ;
      ly ra9              ;
      nc 62   ,hv ra20    ;hvis ikke 62 hop til læsning
[
```

```
b a3                                ;ellers dan gammel kanal 0 læseprogram
[-2]   arn ra        IRC t1         ;
       gr 33         MRA t1         ;
       hv r-2        NRB            ;
       arn ra1       ,pi 2          ;
       gr 28         MA             ;
       arn ra3       ,gr 9          ;
       pmn ra2                      ;
a:     hh 35         X              ;
[34]   pmn 1.3       XD IZA         ;resten af blokken er konstanter
[35]   tl -7         ,ly r1         ;
a3:                                 ;
[36]   pi            LZA t508       ;
[37]   xr            X IZB          ;
[38]   hv 35         LZB            ;
[39]   grf 41        MRC t-1        ;
a1:    arn 999       ,hv 995        ;
a2:    qq 62.16+17.23+14.37+1.38     ;de tre allerede læste karakterer
e                                   ;

e38=200,e39=240,a4=280              ;definer kanal buffere

a20:    pa ra19       ,pa ra13      ;sum og antal=0
a9:     pi            ,hs ra        ;læs etikette
a8:     pm ra5        ,ga ra3
        ga rc6        ,tk 10
        ga ra2        ,tk 10
        gr a4         X MRA
        hv ra6        LRB           ;hop ikke overspringelse
a23:    bt 51         V NQB t-1     ;hvis startetikette tæl melleslag
c6:     pp p[ac]      ,hv ra18      ;ellers tæl i p og hop
        ly ra9        ,hv r-2
e44:    nt 38         ,qq (ra2)
a18h:   pp (ra3)      ,it -2
        pi 1          ,bs p-39
        pp p-40       ,it 1
        can (ra2)     ,hv ra10      ;hop hvis samm kanal
        it (re44)     ,vk (ra2)
c4:     lk (c3)       ,arn ra2
        hs rc1        ,hv ra12

c:      gs r4c        ,ps (re44)
[1c]    bs s-831e88   t-832e88
        vk s1         ,lk (rc3)
        pt r1c        t511
[4c]    ps            ,hh s

a5:     hsf 2
a7:     hs ra         NB            ;læs et ord
a14:    gr se39       MRC           ;gem et ord
        hk rc         ,pp p1
        bs p472       ,hv rc2       ;hop hvis ikke ny kanal
        hs rc         ,pp
a6h:    hs rc1        ,it -1
a2:     btf [ak]      ,it 40        ;tæl kanaler
c2h:    qq (ra3)      ,ps p
a3:     bt [ac]       IRCt-1        ;tæl celler
a12h:   hv ra7        ,ga re44
a10:    hv ra20       NQB
        arn a4        IRA
        hh ra8        NZ
        hv ra9        NRA
[
```

```
        lyn ra16      ,tk -7        ;læs sum og antal
        ly ra16       ,tk-7
        ly ra16       ,tk4
        nc (ra19)     ,hv ra21
        tk 10         ,ca (ra13)
        hs rc1        ,hv ra17

a21:    ps            ,vy 17        ;fejlreaktion
        sy 29         ,sy 64
        sy 18[s]      ,sy 20[u]
        sy 36[m]      ,arn r3
        gr sb13       ,ly r
        vy 32         ,hvn sb13
        gm s3         M t-1

c1:     pt r1c        IRB t-832e88  ;start skrivning paa kanal
        it (rc3)      ,pa ra14
c3:     nt e38        ,qq e38+e39
        vk (re44)     ,it 1
        is (re44)     ,it s+511
        bs 832e88     ,sk (rc3)
        hh s

a:      pmn 1.3       XD IZA        ;læs et ord
        tl -7         ,ly ra16
a16:    pi            LZAt-516
a13:    ac            DX
        hv r1a        XLZ
a19:    xr            t6
        tl 3          ,hr s1
a17:
e

b a12                       ;læs udhopsadresse mm

a:    lyn       D           ;
      nc 14     ,hv ra      ;skip til _
      lyn ra    ,ca 35      ;l
      vy 16     ,hv ra      ;
      ca 17     ,hv e       ;≤
      ca 20     ,hv ra8     ;u
      ca 53     ,hv ra9     ;e
      vy 17     ,sy 64      ;s
      sy 58     ,hv ra      ;

a8:   hsn ra10  X           ;u læs adresse
      hv ra                 ;

a10:  it ra6    ,pt ra4     ;adresselæsning
a1:   lyn ra2   ,ga ra2     ;
      ca        ,hv ra1     ;mellemslag
      ca 32     ,sc ra12 ;-
      ca 16     ,hvn ra7    ;nul
a4:   ca 64     ,hv         ;vr
a2:   bs        t9          ;
      hv ra1                ;ikke ciffer
a7:   it ra3    ,pt ra4     ;ciffer
      mt ra12   ,ml ra5     ;
      tl 39     X NZ        ;
      hv ra1                ;
a3:   xr        ,ga ra11    ;gem adresse
a6:   hv s1                 ;

a5:   10                    ;
a12:  qq                   ;
[
```

```
b a2                    ;

a9:   hsn ra10  X       ;e læs adresse
b14:  vk 960    ,vk 319e88;hent ferritlagerbillede
a:    lk 1000   ,it 1     ;
      vk 293e88 ,it 40    ;
      lk -40    ,it -1    ;
      bt 23     ,hh ra    ;
a1:   arn r a2    t1 IRC    ;sæt celle 6 til 9
      gr 5      t1 MRC    ;
      bt 3      t-1       ;
      hv ra1              ;
a2:   vk 318e88 ,hv 6     ;

      lk 960    ,vk       ;
      qq                  ;
      vy 17               ;
a11:  qq        ,hv (r)   ;
e                         ;
e                         ;
b a2                      ;

b8:   pa ra     t1.3      ;
a:    pmn 1.3   XD IZA    ;
a2:   tl -7               ;
      ly ra1              ;
a1:   pi        LZA t-516 ;
      ac        XD        ;
      hv ra2    X LZ      ;
      xr        t6        ;
      tl 3                ;
      gr 960b8-b14MRC t-1 ;
      bt b8-1b11t-1       ;
      hv ra               ;
b16:  hs 960b11-b14       ;

b7:   tl 10               ;
      gr s5b7-b8M t-1     ;
      hv s                ;

      gm s4     M t-1     ;
      hv s                ;

b: b12=2a1-b7, b13=3b8-b7, b15=4a1-b7                    ;
b6=b-b8, b9=b6+b6+b6+b6, b2=511b7-b, b1=b2-b9           ;
b9=b9+b6+b6, b4=511b8-b7, b5=-512b9, b3=b5-b6, b10=-b9+1;
e                                                        ;
e10                                                      ;SUM CODE i
```

```
b a10, b10                ;
i=10                      ;
     vk  960   , vk  0    ; START: read track 0;
     lk  b     , vy  33   ;   select(p, t);
     pa  b2    t   1      ;   chars:= 1;
     ly        D          ;   typechar;
     sy  14    , sy  17   ;   punch head;
     pt  -1    , vk  0    ;   sum:= 0;
a2h: pp  0     , pmn pb10 ;   for p:= 0 step 1 until 9 do
     ar  1.1   D       LA ;   begin
     ar  1.2   D       LB ;
b2:  qq  1     Xt  6      ;     chars:= chars + 6;
a1h: cl  32    , ga  r1   ;     R:= boot[p]; M:= marks.2;
     sy  -1    , it  -100 ;
     bt  0     , hv  r2   ;     punch the 6 characters;
     cl  -7    , hh  a1   ;
     pa  r-2   , pp  p1   ;
     ncn p-10  ,hh  a2    ;   end;

a3h: pp  0     , pmn pb1  ;   for p:= 0 step 1 until 41 do
     ar  1.1   D       LA ;   begin
     ar  1.2   D       LB ;
     qq (b2)   t   7      ;     chars:= chars + 6;
     ck  3     X          ;     R:= code[p]; M:= mark.39;
     cl  34    , ps  -6   ;
a:   ck  -4    , ga  r2   ;
     bs  s1    , it  64   ;     punch the 7 characters,
     sy  -1    , ps  s1   ;     the last with a 64-hole;
     bs  s     , hv  a4   ;
     cln -6    , hv  a    ;
a4:  pp  p1               ;
     ncn p-42  , hh  a3   ;   end;

     sy  64    , it (b2)  ;   punch tail mark
     pa  -1    , arn -1   ;
     ck  10    , ga  r1   ;   and chars, sum
     sy  -1    , ck  -7   ;
     ga  r1    , ck  -7   ;
     sy  -1    , ga  r1   ;
     sy  -1    , hv  10   ;   goto start;

[Bootstrap program, 6 7-bit characters / word.     Bits of: Curr instr, prec.]
b10:   qq   2.6+ 0.13+ 0.20+  0.27+ 5.34+ 4.39  ; hv s           25-39, length
               [bits 29-34, 36-38 is length]
       qq   2.6+ 4.13+ 0.20+  0.27+ 0.34+ 7.39  ; gm s7 t-1 M    25-39, 0-24
       qqf 28.6+ 2.13+ 0.20+127.27+63.34+11.39  ; hv s     IKC   19-39, 0-24
       qq  32.6+ 6.13+56.20+ 64.27+ 0.34+ 0.39  ; gr xx t1 MPC   25-39, 0-18
[4b10] qq   0.6+ 0.13+58.20+ 32.27+64.34+10.39, ; pi 0 t -49     31-39, 0-24
               [bits 15-20, 22-24 is 510 - length]
       qq  28.6+ 0.13+ 7.20+103.27+49.34+ 8.39  ; hv xx    MR    19-39, 0-30
[6b10] qq   0.6+32.13+64.20+ 58.27+56.34+ 0.39  ; tl 12 V        25-39, 0-18
               [bits 22-27, 29-31 is 513 - length]
       qqf 24.6+44.13+ 1.20+ 32.27+64.34+ 2.39  ; bs (r4)  IO    19-39, 0-24
       qq   0.6+ 0.13+32.20+ 64.27+ 0.34+ 0.39  ;                fill , 0-18
       qq             64.20+  0.27+ 0.34+ 0.39  ;                fill

b1:  vk  960   , vk  0    ; code: select and write track 0;
     sk  r1    , hv  r-1  ;       goto code;
b:   e 10                 ; track 0 is read to here;
```